



SCHOOL OF COMPUTING & INFORMATION TECHNOLOGY

A Project Report On Hyperspectral Satellite Image Classification Using Deep Learning

Submitted in fulfilment of the requirements for the award of the Degree of

Bachelor of Technology

Submitted by

Smitha N - R16CS401

SriPriya M - R16CS411

Srivatsa B - R16CS414

Suma MV - R16CS420

Under the guidance of

Dr. Mallikarjun M Kodabagi

May 2020

**Rukmini Knowledge Park, Kattigenahalli, Yelahanka,
Bengaluru-560064**

www.reva.edu.in

SCHOOL OF COMPUTING & INFORMATION TECHNOLOGY

CERTIFICATE

*This is to certify that the technical project entitled “Hyperspectral Satellite Image Classification Using Deep Learning” is a bonafide work carried out by **SMITHA N (R16CS401)**, **SRI PRIYA (R16CS411)**, **SRIVATSA B (R16CS414)**, and **SUMA MV (R16CS420)** in partial fulfilment for the award of Degree of Bachelor of Technology in Computer Science & Engineering of REVA University, Bangalore, during the year **2019-20**.*

<i>Name & Signature of the Guide</i>	<i>Signature of the Director & Seal</i>
(DR. MALLIKARJUN M KODABAGI)	(Dr. SUNILKUMAR S. MANVI)

Name of the examiners

<i>Examiner - 1</i>		
<i>Examiner – 2</i>		

ABSTRACT

Hyperspectral images are used to give insufficient spectral statistics to accept and differentiate between spectrally different objects. To identify the desired pattern or the object from the image optical analysis techniques are used. Classification of Hyperspectral image is an enormous field of study and huge number of recent advances towards the development has improved the performance for particular applications that uses both the spatial and the spectral image data. The main aim of this Hyperspectral imaging is to find the spectrum for every single pixel within the image, to recognize the processes and to recognize the required materials or the objects. In this study, we are using deep learning framework for the classification of Hyperspectral satellite images. The framework used in this particular study consists of inception module which includes 1x1, 3x3, 5x5 Convolutional layers that give overall classification accuracy of 97.30%.

ACKNOWLEDGEMENT

Any given task achieved is never the result of efforts of a single individual. There are always a group of people who play vital role in leading a task to its completion. Our joy at having successfully finished our project work would be incomplete without thanking everyone who have helped us out through the way. We would like to express our sense of gratitude to our REVA University for providing us the means of attaining our most cherished goal.

We would like to thank our Honourable Chancellor, **Dr. P. Shyama Raju**, Vice-Chancellor, **Dr. Surendra Rao Shankapal**, for their immense support towards students to showcase innovative ideas.

We cannot express enough thanks to our respected Director, **Dr.Sunilkumar S. Manvi** for providing us with highly conducive environment and encouraging the growth and creativity of each and every student. We would also like to offer our sincere gratitude to our Project Co-ordinators, for the numerous learning opportunities that have been provided.

We would like to take this opportunity to express our gratitude to our Guide, **Dr. Mallikarjuna M Kodabagi** for continuously supporting and guiding us in our endeavour as well for taking a keen and active interest in the progress of every phase of our project. Thank you for providing us with the necessary inputs and suggestions for advancing with our project work. We deeply appreciate the wise guidance that sir has provided.

Finally, we would like to extent our sincere thanks to all the faculty members, staff from School of Computing and Information Technology.

SMITHA N

SRI PRIYA M

SRIVATSA B

SUMA MV

TABLE OF CONTENTS

Chapter	Page No.
1. Introduction	viii
1.1 Objectives	ix
2. Literature Survey	x
3. System Design and Analysis	xii
3.1 Data	xii
3.2 Feature Extraction	xii
3.3 Classification	xii
3.4 Classified Output	xii
4. System Requirements	xiii
4.1 Tensorflow	xiii
4.2 Keras	xiii
4.3 Sklearn	xiii
5. Methodology	xiv
5.1 Data Description	xv
5.2 Dimensionality Reduction of the Input Image	xvi
5.3 Creation of Input Patches	xvi
5.4 Classification Model	xvii
6. Source Code	xviii
6.1 Dataset Creation	xviii
6.1.1 Dataset Used	xviii
6.1.2 Loading the Dataset	xix
6.1.3 Dimensionality Reduction	xxi
6.1.4 Creating training and testing data	xxv

6.2	Training the classification model	xxviii
6.2.1	Loading the training data	xxviii
6.2.2	Defining the classification model	xxix
6.2.3	Training	xxx
6.3	Validation	xxxii
7.	Experimental Results	xxxiv
8.	Conclusion and Future Scope	xxxviii
9.	References	xxxix
10.	Appendices	
	Appendix A: Registration fee paid receipt	xli
	Appendix B: Conference paper presentation	xl ii
	Certificate	
	Appendix C: Paper	xl viii
	Appendix D: Plagiarism check certificate	liii

LIST OF FIGURES

Figure No.	Figure Caption	Page No.
1	System Overview	xii
2	Block Diagram of Overall System Architecture	xiv
3	Hyperspectral Image	xv
4	Ground Truth Image	xv
5	Statistical Analysis of Classification	xxxiv
6	Confusion Matrix Obtained During Classification	xxxv
7	Classification Map Obtained After Applying Classification Model on Indian Pines Data Set	xxxvii

LIST OF TABLES

Table No.	Table Caption	Page No.
1	Ground truth classes for the Indian Pines scene and their respective samples numbers	xviii
2	Accuracy Metrics	xxxvi

CHAPTER 1: INTRODUCTION

Remote sensing is the science of obtaining information about objects or areas from a distance, typically from aircraft or satellites. Remote sensors collect data by detecting the energy that is reflected from Earth. These sensors can be on satellites or mounted on aircraft.

Remote sensors can be either passive or active. Passive sensors respond to external stimuli. They record natural energy that is reflected or emitted from the Earth's surface. The most common source of radiation detected by passive sensors is reflected sunlight. In contrast, active sensors use internal stimuli to collect data about Earth. For example, a laser-beam remote sensing system projects a laser onto the surface of Earth and measures the time that it takes for the laser to reflect back to its sensor.

Remote sensing techniques are the techniques where information is obtained by indirect computation of the thing under study. The data which is obtained is via non-particulate radiation.

Remote sensing covers huge areas that enable surveys of geographical area for identification of immensely large features. It allows repeated coverage which are very useful when collected data on different themes like agricultural fields, forests, water and so on.

Hyperspectral images are volumetric cubes that consist of large spatial images. Each spatial image or band captures the ground objects at a defined wave length. Due to rich spectral information obtained through these images, hyperspectral imagery has become an important tool for remote sensing and scanning applications. Hyperspectral images are widely employed in the sensing and discovering of minerals, military surveillance and give information about how to monitor the Earth's resources.

The technique is so sensitive that it can detect camouflaged objects and has been used in forestry to measure biomass and damage caused by plant disease. Hyperspectral remote sensing combines imaging and spectroscopy in a single system, which often includes large data sets and require new processing methods.

Hyperspectral image (HSI) classification is a phenomenal mechanism to analyse diversified land cover in remotely sensed hyperspectral images. Given a set of

observations with known class labels, the basic goal of hyperspectral image classification is to assign a class label to each pixel.

We are using Inception module approach to perform classification. Using this module, we classify Indian pines Hyperspectral images and extract desired features more accurately and efficiently.

1.1 Objectives:

Objective of Hyperspectral data handling is to differentiate collected images into distinctive materials or the objects having related to particular applications, and produce classification details that indicate where the materials or the objects are present. The corresponding response can help us figuring out how land-cover maps for environmental remote sensing can be used to mineral mapping for geological applications, mineral exploration, finding vegetation areas for agricultural studies or for urban mapping.

We can even compare the images from the past to the present and check for the changes like that we can make prediction of how the land is going to change based on weather conditions so that we get to know where and how the urban planning must be done and where the vegetation should be developed.

CHAPTER 2: LITERATURE SURVEY

Lokman G. & Yilmaz, G. It is paper on hyperspectral image classification where they use SVM [2], here the model uses the vector neural network for Hyperspectral image. In this project the input given to the system is pre-processed hyperspectral image data. It contains of the image that are classified in matrix form as the result of the system. The output of the background data with other values and the target class is in negative values. To test the model, they used Hyperion sensor on NASA EO-1 satellite to classify the airborne visible/infrared imaging spectrometer (AVIRIS) image of Okavango Delta and image of Salinas Valley.

In this paper [3], IdoFaran, Nathan S. Netanyahu, Maxim Shoshany, FadiKizel, Eli (Omid) David, JisungGeba Chang, RonitRud have proposed a new method, to beat the problem of lack of sufficient data by training a deep neural network for classification. To train the convolutional neural network for pixel-based classification, they used parallel ground truth from hyperspectral image with high resolution.

[4] In this paper to increase the performance of the hyperspectral image classification based on Siamese convolution neural network (S-CNN) using deep learning for feature extraction method, they used samples that are labelled for the training. Classification was done by training the S-CNN [5]. The feature of hyperspectral cube is derived from five-layer Convolution neural network of nonlinear transformation function. The classification task is performed by using supervised margin ranking loss function. Thus, the support vector machine gives the better classification performance when compared to conventional method.

In this paper [6], Lu, Y., Perez, D., Dao, M., Kwan, C., & Li, J. have brought the images from WorldView-2 satellite for soil observation by combining original 8 multispectral bands and 80 synthetic hyperspectral bands. The CNN model here used for soil detection on the curved surface area has increased its average by 7.42%. They have also mentioned about how the soil detection performance is increased by

pan-sharpening and morphological post processing. For improving the performance of classifying the object detection and remote sensing application synthetic hyperspectral bands of CNN model is used.

In this paper [7], Shen, Y., Xiao, L., Chen, J., & Pan, D. have used deep ELM neural network for Indian Pines dataset, Salinas dataset for testing and Pavia University dataset. This paper explains the two part of frameworks that are used in their project i.e. fully connected deep ELM network and two branch convolutional learning module within hidden nodes. They have extracted random weight of Hyperspectral image spectral and spatial features, the extracted data is concentrated and fed into the fully connected ELM networks. The classified result is obtained from ELM network.

In this paper [8], AVIRIS and ROSIS hyperspectral datasets is used to developed framework of deep learning and is used for validation and experimentation. The spectral and spatial information are encoded using high-level features of MLPs and CNN. They have compared the performance and accuracy rate between SVM based classifier and deep learning approach. For every data set given for the validation, deep learning approach has the superior performance when compared to others.

CHAPTER 3: SYSTEM DESIGN

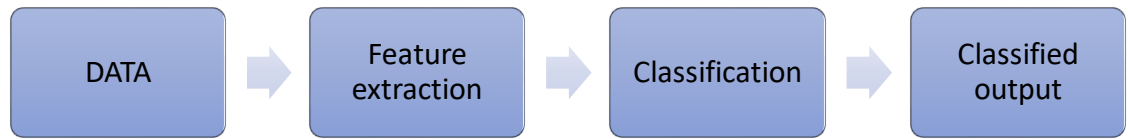


FIGURE 1: SYSTEM OVERVIEW

3.1 Data:

Satellite Data: Satellite images are images of Earth collected by imaging satellites operated by governments and businesses around the world.

Hyperspectral Images: Hyperspectral images or Hyperspectral imaging, like other spectral imaging, collects and processes information from across the electromagnetic spectrum. The goal of hyperspectral imaging is to obtain the spectrum for each pixel in the image of a scene, with the purpose of finding objects, identifying materials, or detecting processes.

3.2 Feature Extraction:

Feature extraction is the transformation of original data to a data set with a reduced number of variables which contains the most discriminatory information and it selects a subset of relevant feature for model construction, thus reduces training time, simplifies the models and improves the classification accuracy.

3.3 Classification:

The classification process is to categorize all pixels in a digital image into one of several land cover classes, or "themes". This categorized data may then be used to produce thematic maps of the land cover present in an image. The objective of image classification is to identify and portray, as a unique grey level (or colour), the features occurring in an image in terms of the object or type of land cover these features actually represent on the ground.

3.4 Classified Output:

A classified image with each pixel classified into its respective predefined class labels is obtained as the final output. Once the classification is done, we get the classified map for the given ground truth image.

CHAPTER 4: SYSTEM REQUIREMENTS

4.1 Tensorflow:

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

TensorFlow allows developers to create dataflow graphs—structures that describe how data moves through a graph, or a series of processing nodes. Each node in the graph represents a mathematical operation, and each connection or edge between nodes is a multidimensional data array, or tensor.

4.2 Keras:

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible.

Keras was created to be user friendly, modular, easy to extend, and to work with Python. The API was “designed for human beings, not machines,” and “follows best practices for reducing cognitive load.”

The biggest reasons to use Keras stem from its guiding principles, primarily the one about being user friendly. Beyond ease of learning and ease of model building, Keras offers the advantages of broad adoption, support for a wide range of production deployment options, integration with at least five back-end engines (TensorFlow, CNTK, Theano, MXNet, and PlaidML), and strong support for multiple GPUs and distributed training. Plus, Keras is backed by Google, Microsoft, Amazon, Apple, Nvidia, Uber, and others.

4.3 Sklearn:

Sklearn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

CHAPTER 5: METHODOLOGY

The below figure (Figure 2) represents the architecture of our system,

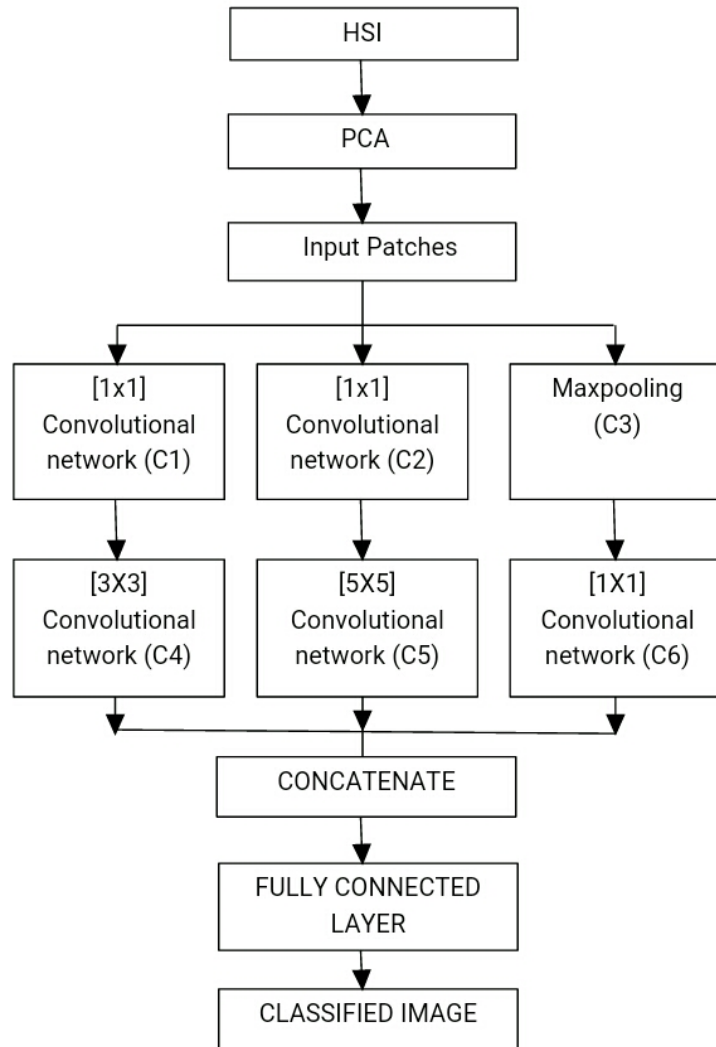


FIGURE 2: BLOCK DIAGRAM OF OVERALL SYSTEM ARCHITECTURE

5.1. Data description:

In this method of classification, the satellite image i.e. hyperspectral satellite images (HSI) are provided as the input. These images consist of n number of bands/channels which has spectral and spatial information of each and every pixel. By considering a single hyperspectral image (Figure 3) and its ground truth image (Figure 4) as input data, classification process is applied to classify each pixel into its respective classes. Ground truth data of the image gives us information about the various predefined classes of the pixels.



FIGURE 3: HYPERSPECTRAL IMAGE

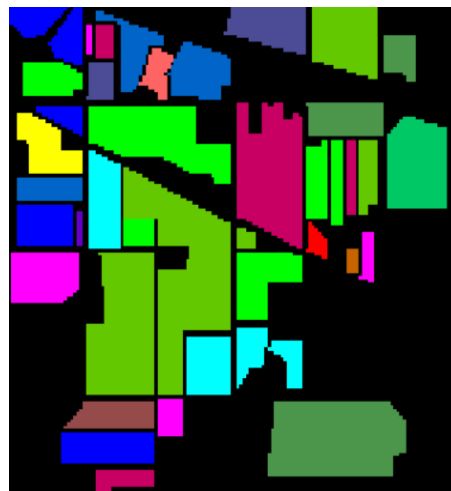


FIGURE 4: GROUND TRUTH IMAGE

The hyperspectral image is represented as $W \times H \times B$ dimensional tensor, where W , H & B is width, height of the image and number of spectral bands/channels respectively, consisting of lot of spectral and spatial information.

5.2. Dimensionality reduction of the input image:

The input image containing hundreds of spectral bands lead to high dimensionality which may result in more training time and high computational costs. From an analysis made on spectral response patterns of pixels, it is observed that the pixels which belong to same class have a very small variance in their response patterns. This predicts that these pixels have similar properties and almost the same pixel values in every channel, while the pixels belonging to different classes have different properties due to large amount of variance. Taking this into account dimensionality reduction is achieved by employing PCA (Principal Component Analysis) [9] approach along the spectral dimension to abridge the entire image into lower dimension. PCA reduces few spectral properties which possess less variance by preserving the spatial properties.

In this experimentation process the first 30 principle components are retained to preserve most of the initial information. After the application of PCA, the input image of dimension $W \times H \times B$ is reduced to $W \times H \times B_r$, where B_r is the number of bands after the application of PCA.

5.3. Creation of Input patches:

The hyperspectral image with reduced dimensions after applying PCA is further decomposed into numerous patches of dimension $S \times S \times B_r$ which are also 3 Dimensional tensors. Here S corresponds to length of the square patch. To classify a pixel at location (x, y) to the corresponding class label $L(x, y)$, a patch $P(x, y)$ of dimension $S \times S \times B_r$ centred at that pixel is used. Hence the dataset contains pixel's information in the form $D = \{P(x, y), L(x, y)\}$ where $x = 1, 2, \dots, W$ and $y = 1, 2, \dots, H$.

S is kept minimum as 5 in order to have the neighboring pixels as minimum as possible during the classification process which may otherwise increase the computational cost of training data.

5.4. Classification model:

After applying PCA with reduced dimensions we get a input patch which is sent to the deep learning classification structure and 25% of the dataset is used for testing and remaining 75% of data is used to train the classification model. The model has to be trained for all kinds of samples with highest accuracy. The Convolutional Neural Network (CNN) architecture that is used here is similar to the inception module. It has convolutional layers C1, C2 along with trainable filters of dimension 1 x 1 and one more layer C3 as maxpooling layer. The C1 and C2 convolutional layers are followed by another set of convolutional layers C4 and C5 with 60 filters which are trainable, the corresponding dimension of filters are 3x3 and 5x5. Later 1x1 convolution layer is added before 3x3 5x5 convolutional layers, while considering the height and width of the feature map, the number of computations is decreased by a factor of 10, this reduces the computational requirements and in turn maximises the efficiency. C3 being maxpooling layer is followed by C6, the convolution layer of dimension 1x1.

The output from each of C4, C5 and C6 layers is combined together to form the output of the inception module. Later the output obtained is flattened into one dimensional set of neurons which is used to create a fully connected neural network layer for final classification. Our model was trained for 25 epochs. The training accuracy of the model increased for every epoch. A classified image with each pixel classified into its corresponding class labels is obtained as the final output.

CHAPTER 6: SOURCE CODE

6.1. Dataset Creation:

6.1.1.Dataset used:

In our experiment the Indian Pines data set captured by the AVIRIS sensor over the Indian Pines test site in North-western Indiana is used. It has 224 spectral reflectance bands of wavelengths ranging from 0.4 to 2.5 10^{-6} meters and 145x145 pixels. In this experiment 75% of data set sample is selected as training dataset and remaining 25% as testing dataset. The Indian Pines data consists of agriculture and forest/others natural vegetation. The ground truth image of the dataset is differentiated into 16 classes, each class having a particular number of samples as shown in Table 1 below.

Table 1: Ground truth classes for the Indian Pines scene and their respective samples numbers

#	Class	Samples
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93

The Indian Pines dataset used here is available in the form of .mat files which store the image in the form of NumPy arrays.

- Indian_pines_corrected.mat: contains hyperspectral image containing 200 bands.
- indian_pines_gt.mat: contains groundtruth information of the hyperspectral image.

6.1.2. Loading the Dataset:

The hyperspectral image is loaded into 'X' and its corresponding class label information in ground truth image is loaded into 'y'.

```
def loadIndianPinesData():  
    data_path = os.path.join(os.getcwd(), 'Data')  
    data = sio.loadmat(os.path.join(data_path, 'Indian_pines_corrected.mat'))['indian_pines_corrected']  
    labels = sio.loadmat(os.path.join(data_path, 'indian_pines_gt.mat'))['indian_pines_gt']  
    return data, labels  
  
X, y = loadIndianPinesData()
```

The initial dimensions of the image:

```
X.shape
```

```
(145, 145, 30)
```

```
y.shape
```

```
(145, 145)
```

Image represented as NumPy array:

```
In [5]: print(X)

[[[3172 4142 4506 ... 1057 1020 1020]
  [2580 4266 4502 ... 1064 1029 1020]
  [3687 4266 4421 ... 1061 1030 1016]
  ...
  [2570 3890 4320 ... 1042 1021 1015]
  [3170 4130 4320 ... 1054 1024 1020]
  [3172 3890 4316 ... 1043 1034 1016]]

[[[2576 4388 4334 ... 1047 1030 1006]
  [2747 4264 4592 ... 1055 1039 1015]
  [2750 4268 4423 ... 1047 1026 1015]
  ...
  [3859 4512 4605 ... 1056 1035 1015]
  [3686 4264 4690 ... 1051 1012 1020]
  [2744 4268 4597 ... 1047 1019 1016]]

[[[2744 4146 4416 ... 1055 1029 1025]
  [2576 4389 4416 ... 1051 1021 1011]
  [2744 4273 4420 ... 1068 1033 1010]
  ...
  [2570 4266 4509 ... 1051 1025 1010]
  [2576 4262 4496 ... 1047 1029 1020]
  [2742 4142 4230 ... 1042 1025 1011]]

...

[[[3324 3728 4002 ... 1003 1004 1004]
  [2983 3604 3829 ... 1011 1013 1008]
  [2988 3612 3913 ... 1012 1001 1004]
  ...
  [2564 4115 4103 ... 1003 1005 1013]
  [2730 4111 4103 ... 1015 1013 1004]
  [3156 3991 4103 ... 1017 1014 1000]]]
```

```

[[3161 3731 3834 ... 1002 1000 1000]
 [2727 3742 4011 ... 999 991 1003]
 [2988 4114 4011 ... 1006 1008 1013]
 ...
 [3156 3858 4016 ... 1011 1004 1003]
 [3159 3858 4100 ... 1016 1000 1000]
 [2561 3866 4003 ... 1008 1008 1000]]

[[2979 3728 3732 ... 1006 1004 1000]
 [2977 3728 3741 ... 1007 1009 990]
 [2814 3728 3914 ... 999 1009 1003]
 ...
 [3153 3864 4282 ... 1003 1008 1000]
 [3155 4104 4106 ... 1011 1005 1003]
 [3323 3860 4197 ... 1007 1004 1000]]]

```

```
In [6]: print(y)
```

```

[[3 3 3 ... 0 0 0]
 [3 3 3 ... 0 0 0]
 [3 3 3 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]

```

6.1.3 Dimension reduction:

Initially we set the following parameters:

```

# Load the Global values (windowSize, numPCComponents, testRatio) from the text file global_variables.txt
myFile = open('global_variables.txt', 'r')
file = myFile.readlines()[:]

for line in file:

    if line[0:3] == "win":

        ds = line.find('=')
        windowSize = int(line[ds+1:-1],10)

    elif line[0:3] == "num":

        ds = line.find('=')
        numPCComponents = int(line[ds+2:-1],10)

    else:

        ds = line.find('=')
        testRatio = float(line[ds+1:])

# Global Variables
#numPCComponents = 30
#windowSize = 5
#testRatio = 0.25

```

- The length of the square patch is set to 5 in order to take into consideration the closest 24 neighbours of each pixel.
- The first 30 principal components are used here, to preserve 99.9% of information and reducing this way up to 15 times the dimensionality of the raw input.
- The test ratio is set to 0.25 so that 75% data is being used for training and remaining 25% for testing.

Applying PCA:

```
def applyPCA(X, numComponents=75):
    newX = np.reshape(X, (-1, X.shape[2]))
    pca = PCA(n_components=numComponents, whiten=True)
    newX = pca.fit_transform(newX)
    newX = np.reshape(newX, (X.shape[0], X.shape[1], numComponents))
    return newX, pca

X, pca = applyPCA(X, numPCAcomponents)
```

The hyperspectral data fed to 'X' having 200 channels is reduced to a dimension of 30 channels.

The dimensions of the image after applying PCA:

```
In [14]: X.shape
```

```
Out[14]: (145, 145, 30)
```

```
In [15]: y.shape
```

```
Out[15]: (145, 145)
```

This image is further disintegrated into square patches of size 5x5:

```
def createPatches(X, y, windowSize=5, removeZeroLabels = True):
    margin = int((windowSize - 1) / 2)
    zeroPaddedX = padWithZeros(X, margin=margin)
    # split patches
    patchesData = np.zeros((X.shape[0] * X.shape[1], windowSize, windowSize, X.shape[2]))
    patchesLabels = np.zeros((X.shape[0] * X.shape[1]))
    patchIndex = 0
    for r in range(margin, zeroPaddedX.shape[0] - margin):
        for c in range(margin, zeroPaddedX.shape[1] - margin):
            patch = zeroPaddedX[r - margin:r + margin + 1, c - margin:c + margin + 1]
            patchesData[patchIndex, :, :, :] = patch
            patchesLabels[patchIndex] = y[r-margin, c-margin]
            patchIndex = patchIndex + 1
    if removeZeroLabels:
        patchesData = patchesData[patchesLabels>0, :, :, :]
        patchesLabels = patchesLabels[patchesLabels>0]
        patchesLabels -= 1
    return patchesData, patchesLabels

XPatches, yPatches = createPatches(X, y, windowSize=windowSize)
```

- XPatches: contains the list of patches into which a single hyperspectral image is disintegrated.
- yPatches: contains the list of corresponding ground truth class labels.

The size of the patches created is:

```
XPatches.shape
```

```
(10249, 5, 5, 30)
```

```
yPatches.shape
```

```
(10249,)
```

- Hyperspectral Image Patches of size 5x5 having 30 channels each are created.
- There are total 10249 patches of samples created.

The dataset samples stored as NumPy arrays:

XPatches:

```
array([[[[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00]],

       [[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00]],

       [[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 9.68768490e-01,  4.80151196e-01,  9.50128450e-02, ...,
          -3.59242138e-01, -1.36133869e-02, -1.86586966e-01],
        [ 1.08206297e+00, -6.66886184e-01,  4.57615428e-01, ...,
          -4.66595699e-01, -1.67086706e+00,  3.59427782e+00],
        [ 1.11968468e+00, -1.01852852e+00,  6.41120878e-01, ...,
          1.82460714e-01,  7.57366727e-01, -1.13213920e-01]],

       [[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
          0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
        [ 8.87736424e-01,  7.79240148e-01, -3.90661325e-02, ...,
          -1.67043508e+00, -7.48645820e-02, -3.38973942e-01],
        [ 1.00143995e+00, -2.56748297e-01,  2.37507652e-01, ...,
          1.69436957e-01,  1.59501219e-01, -2.71913166e-01],
        [ 1.05092573e+00, -7.09482486e-01,  3.92527636e-01, ...,
          -2.68184738e-01,  3.46418924e-01, -2.16884309e-01]]],
```



```

.../

[[ 7.81909246e-01, -6.72547756e-02, -8.26451179e-01, ...,
   3.73361805e-01,  3.86787215e-01,  1.11821701e-01],
 [ 8.48429862e-01, -1.41022491e-01, -7.88488104e-01, ...,
   4.06196746e-01,  5.07536104e-01,  6.89622555e-03],
 [ 9.09078769e-01, -1.51209638e-01, -7.63382845e-01, ...,
   6.13936860e-01,  5.47626367e-01, -8.95294875e-02],
 [ 6.01372271e-01, -2.50140546e-01, -1.13320862e+00, ...,
   2.79924616e-01, -1.53241435e-01, -2.80723535e-01],
 [-4.04699945e-01, -3.75608691e-02, -7.67774885e-01, ...,
   1.44439680e+00,  9.07874659e-01,  1.03495971e+00]],

[[ 7.89528453e-01,  2.81449625e-02, -8.63337934e-01, ...,
   1.38854784e-01,  4.92455609e-01,  3.14549450e-02],
 [ 8.07582034e-01,  6.49356326e-03, -8.71115309e-01, ...,
   1.05117615e-01,  9.11390045e-01,  3.19396256e-01],
 [ 8.40581378e-01,  7.35883021e-04, -8.89890734e-01, ...,
   1.83207367e-01, -1.56418860e+00,  3.62365776e+00],
 [ 5.68373882e-01, -1.54782724e-01, -1.00658421e+00, ...,
   -4.91516037e-02,  3.89904123e-01, -4.54070623e-01],
 [-5.55191572e-01, -3.95878833e-01, -1.14346407e+00, ...,
   9.78496106e-01,  4.71057620e-01, -3.63421666e-01]],

[[ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
   0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
 [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
   0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
 [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
   0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
 [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
   0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
 [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
   0.00000000e+00,  0.00000000e+00,  0.00000000e+00],
 [ 0.00000000e+00,  0.00000000e+00,  0.00000000e+00, ...,
   0.00000000e+00,  0.00000000e+00,  0.00000000e+00]]])

```

yPatches:

```
array([2., 2., 2., ..., 9., 9., 9.])
```

6.1.4 Creating training and testing data:

The dataset created above is split into training and testing data with a split test ratio of 0.25. 75% of the data is used for training and 25% for testing.

```

def splitTrainTestSet(X, y, testRatio=0.10):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=testRatio, random_state=345,
                                                         stratify=y)
    return X_train, X_test, y_train, y_test

X_train, X_test, y_train, y_test = splitTrainTestSet(XPatches, yPatches, testRatio)

```

- X_train: XPatchessplit for training.
- X_test:XPatches split for testing.
- y_train: corresponding yPatches split for training.
- y_test: corresponding yPatches split for testing.

Oversampling the weak classes:

Oversampling duplicates the samples from the minority class in the training dataset in order to balance the dataset samples.

```
def oversampleWeakClasses(X, y):
    uniqueLabels, labelCounts = np.unique(y, return_counts=True)
    maxCount = np.max(labelCounts)
    labelInverseRatios = maxCount / labelCounts
    # repeat for every label and concat
    newX = X[y == uniqueLabels[0], :, :, :].repeat(round(labelInverseRatios[0]), axis=0)
    newY = y[y == uniqueLabels[0]].repeat(round(labelInverseRatios[0]), axis=0)
    for label, labelInverseRatio in zip(uniqueLabels[1:], labelInverseRatios[1:]):
        cX = X[y == label, :, :, :].repeat(round(labelInverseRatio), axis=0)
        cY = y[y == label].repeat(round(labelInverseRatio), axis=0)
        newX = np.concatenate((newX, cX))
        newY = np.concatenate((newY, cY))
    np.random.seed(seed=42)
    rand_perm = np.random.permutation(newY.shape[0])
    newX = newX[rand_perm, :, :, :]
    newY = newY[rand_perm]
    return newX, newY

X_train, y_train = oversampleWeakClasses(X_train, y_train)
```

Augmenting data:

The training patches in the dataset are augmented with more samples by rotating each patch in all the co-ordinate axes directions and flipping the image up-down and left-right.

```

def AugmentData(X_train):
    for i in range(int(X_train.shape[0]/2)):
        patch = X_train[i,:,:,:]
        num = random.randint(0,2)
        if (num == 0):

            flipped_patch = np.flipud(patch)
        if (num == 1):

            flipped_patch = np.fliplr(patch)
        if (num == 2):

            no = random.randrange(-180,180,30)
            flipped_patch = scipy.ndimage.interpolation.rotate(patch, no,axes=(1, 0),
                                                                reshape=False, output=None, order=3, mode='constant', cval=0.0, prefilter=False)

        patch2 = flipped_patch
        X_train[i,:,:,:] = patch2

    return X_train

X_train = AugmentData(X_train)

```

Saving the pre-processed data:

After performing the above pre-processing measures, test and train patches are saved as NumPy arrays into separate NumPy array files(.npz).

```

def savePreprocessedData(X_trainPatches, X_testPatches, y_trainPatches, y_testPatches, windowSize, wasPCAapplied = False, numPCAComponents = 0, testRatio = 0.25):
    if wasPCAapplied:
        with open("X_trainPatches_" + str(windowSize) + "PCA" + str(numPCAComponents) + "testRatio" + str(testRatio) + ".npz", 'bw') as outfile:
            np.save(outfile, X_trainPatches)
        with open("X_testPatches_" + str(windowSize) + "PCA" + str(numPCAComponents) + "testRatio" + str(testRatio) + ".npz", 'bw') as outfile:
            np.save(outfile, X_testPatches)
        with open("y_trainPatches_" + str(windowSize) + "PCA" + str(numPCAComponents) + "testRatio" + str(testRatio) + ".npz", 'bw') as outfile:
            np.save(outfile, y_trainPatches)
        with open("y_testPatches_" + str(windowSize) + "PCA" + str(numPCAComponents) + "testRatio" + str(testRatio) + ".npz", 'bw') as outfile:
            np.save(outfile, y_testPatches)
    else:
        with open("../preprocessedData/XtrainWindowSize" + str(windowSize) + ".npz", 'bw') as outfile:
            np.save(outfile, X_trainPatches)
        with open("../preprocessedData/XtestWindowSize" + str(windowSize) + ".npz", 'bw') as outfile:
            np.save(outfile, X_testPatches)
        with open("../preprocessedData/ytrainWindowSize" + str(windowSize) + ".npz", 'bw') as outfile:
            np.save(outfile, y_trainPatches)
        with open("../preprocessedData/ytestWindowSize" + str(windowSize) + ".npz", 'bw') as outfile:
            np.save(outfile, y_testPatches)

savePreprocessedData(X_train, X_test, y_train, y_test, windowSize = windowSize,
                    wasPCAapplied=True, numPCAComponents = numPCAComponents, testRatio = testRatio)

```

The following numpy array files are created:

- [X_testPatches_5PCA30testRatio0.25.npz](#) : XPatches used for testing.
- [X_trainPatches_5PCA30testRatio0.25.npz](#) :XPatches used for training.
- [y_testPatches_5PCA30testRatio0.25.npz](#) :yPatches used for testing.
- [y_trainPatches_5PCA30testRatio0.25.npz](#) :yPatches used for training.

6.2. Training the classification model:

6.2.1. Loading the training data:

```
X_train = np.load("X_trainPatches_" + str(windowSize) + "PCA" + str(numPCAcomponents) + "testRatio" + str(testRatio) + ".npz")
y_train = np.load("y_trainPatches_" + str(windowSize) + "PCA" + str(numPCAcomponents) + "testRatio" + str(testRatio) + ".npz")

# Reshape into (numberofsamples, channels, height, width)
X_train = np.reshape(X_train, (X_train.shape[0],X_train.shape[3], X_train.shape[1], X_train.shape[2]))

# convert class labels to on-hot encoding
y_train = np_utils.to_categorical(y_train)

# Define the input shape
input_shape= X_train[0].shape
print(input_shape)

(30, 5, 5)

# number of filters
C1 = 3*numPCAcomponents
print(C1)

90
```

- X_train and y_train patches numpy array files are loaded into X_train and y_train.
- The new X_train parameter is reshaped into a numpy array of the respective number of samples, channels, height and width.
- One-hot encoding is applied to y_train categorical data. Categorical data are variables which contain label values for each class rather than numeric values. To convert these label values to binary numbers one-hot encoding is applied.
- The input shape is defined as 30x5x5 (30 channels, width & height – 5).
- We set the number of filters to 90.

6.2.2. Defining the classification model:

Once after loading the training dataset, we define the classification model to perform training.

```
from keras.layers import Input
input_img = Input(shape = (30, 5, 5))

x1=(Conv2D(C1, (1,1), padding='same', activation='relu'))(input_img)
x1 = (Conv2D(C1, (3,3), padding='same', activation='relu'))(x1)
x2 = Conv2D(C1, (1,1), padding='same', activation='relu')(input_img)
x2 = Conv2D(C1, (5,5), padding='same', activation='relu')(x2)
x3 = MaxPooling2D((3,3), strides=(1,1), padding='same')(input_img)
x3 = Conv2D(C1, (1,1), padding='same', activation='relu')(x3)
output = keras.layers.concatenate([x1, x2, x3], axis = 3)
output = Flatten()(output)
out = Dense(16, activation='softmax')(output)
```

- The input patch given to classification model is converted to 3 Dimensional tensor of shape 30x5x5.
- The model consists of Conv2D network layers with different filter sizes.
- The input image tensor is sent to Conv2D layer of 1x1 filter size having ‘relu’ activation function which is further sent to a Conv2D layer of filter size 3x3 (x1).
- The input image tensor is sent to Conv2D layer of 1x1 filter size having ‘relu’ activation function which is further sent to a Conv2D layer of filter size 5x5 (x2).
- The input image is also sent to a maxpooling layer of stride 1x1, which is further sent to a Conv2D layer of 1x1 filter size having ‘relu’ activation function (x3).
- The x1,x2 and x3 outputs obtained are concatenated into single output.
- This concatenated output of is flattened into a single array.
- The final output is obtained by a fully connected layer on which ‘softmax’ activation function is applied.

Model summary:

```
from keras.models import Model
model = Model(inputs = input_img, outputs = out)
print(model.summary())
```

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 30, 5, 5)	0	
conv2d_1 (Conv2D)	(None, 90, 5, 5)	2790	input_1[0][0]
conv2d_3 (Conv2D)	(None, 90, 5, 5)	2790	input_1[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 30, 5, 5)	0	input_1[0][0]
conv2d_2 (Conv2D)	(None, 90, 5, 5)	72990	conv2d_1[0][0]
conv2d_4 (Conv2D)	(None, 90, 5, 5)	202590	conv2d_3[0][0]
conv2d_5 (Conv2D)	(None, 90, 5, 5)	2790	max_pooling2d_1[0][0]
concatenate_1 (Concatenate)	(None, 90, 5, 15)	0	conv2d_2[0][0] conv2d_4[0][0] conv2d_5[0][0]
flatten_1 (Flatten)	(None, 6750)	0	concatenate_1[0][0]
dense_1 (Dense)	(None, 16)	108016	flatten_1[0][0]
Total params: 391,966			
Trainable params: 391,966			
Non-trainable params: 0			
None			

6.2.3. Training:

```
In [14]: from keras.optimizers import SGD
epochs = 25
lr = 0.1
decay = lr/epochs
sgd = SGD(lr=lr, momentum=0.9, decay=decay, nesterov=False)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
model.fit(X_train, y_train, epochs=epochs, batch_size=32)
```

- The model is trained for 25 epochs. An **epoch** in deep learning refers to one cycle through the full training dataset.

There is an increasing in training accuracy in every epoch.

- The quantity of change to the model throughout every step of this search method, or the step size, is termed as the “**learning rate**” and provides maybe the foremost vital hyperparameter to tune the neural network so as to attain good performance on our problem.

- Learning rate is a parameter that defines how much an updating step influences the current value of the weights whereas weight **decay** is a term in the weight update rule that causes the weights to exponentially decay to zero, if no other update is scheduled.
- **SGD (Stochastic Gradient Descent)** is an iterative optimization technique that reduces the number of computations to be performed by randomly picking the required data points out of the given set of data points in each iteration.
- The model is then compiled to deliver best execution performance.
- We first feed the training data(X_train) and training labels(y_train). We then use Keras to allow our model to train for 100 epochs on a batch_size of 32.

The trained model can be saved in the following format:

```
import h5py
from keras.models import load_model
```

```
model.save('my_model' + str(windowSize) + 'PCA' + str(numPCAcomponents) + "testRatio" + str(testRatio) + '.h5')
```

- `my_model5PCA30testRatio0.25.h5` is the file obtained after saving the trained classification model.
- It includes the following:
 - ✓ The model's architecture.
 - ✓ The model's weight values which were learned during training.
 - ✓ The model's training configuration (the parameters passed to compile)
 - ✓ The optimizer and its state, if any (enables us to restart training where it was left)

6.3. Validation:

```
X_test = np.load("X_testPatches_" + str(windowSize) + "PCA" + str(numPCComponents) + "testRatio" + str(testRatio) + ".npy")
```

```
y_test = np.load("y_testPatches_" + str(windowSize) + "PCA" + str(numPCComponents) + "testRatio" + str(testRatio) + ".npy")
```

```
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[3], X_test.shape[1], X_test.shape[2]))  
y_test = np_utils.to_categorical(y_test)
```

```
# load the model architecture and weights
```

```
model = load_model('my_model' + str(windowSize) + 'PCA' + str(numPCComponents) + "testRatio" + str(testRatio) + '.h5')
```

- The X_test and y_test patches stored in numpy array (.npy) format are loaded into the parameters X_test: containing Xpatches for testing.

Y_test: containing ypatches or corresponding class labels of X_test for testing.

- The classification model which was saved earlier during training is loaded into the parameter 'model'.
- Using the pre-trained model we make predictions and print the result as a classification report.
- The classification report consists of different accuracy metrics, precision and score.

```
def reports (X_test,y_test):  
    Y_pred = model.predict(X_test)  
    y_pred = np.argmax(Y_pred, axis=1)  
    target_names = ['Alfalfa', 'Corn-notill', 'Corn-mintill', 'Corn'  
                    , 'Grass-pasture', 'Grass-trees', 'Grass-pasture-mowed',  
                    'Hay-windrowed', 'Oats', 'Soybean-notill', 'Soybean-mintill',  
                    'Soybean-clean', 'Wheat', 'Woods', 'Buildings-Grass-Trees-Drives',  
                    'Stone-Steel-Towers']  
  
    classification = classification_report(np.argmax(y_test, axis=1), y_pred, target_names=target_names)  
    confusion = confusion_matrix(np.argmax(y_test, axis=1), y_pred)  
    score = model.evaluate(X_test, y_test, batch_size=32)  
    Test_Loss = score[0]*100  
    Test_accuracy = score[1]*100  
    oa = accuracy_score(np.argmax(y_test, axis=1), y_pred)  
    each_acc, aa = AA_andEachClassAccuracy(confusion)  
    kappa = cohen_kappa_score(np.argmax(y_test, axis=1), y_pred)  
    return classification, confusion, Test_Loss, Test_accuracy, oa*100, each_acc*100, aa*100, kappa*100
```


Finally we obtain the predicted image using the pre-trained classification model.

```
# calculate the predicted image
outputs = np.zeros((height,width))
for i in range(height-PATCH_SIZE+1):
    for j in range(width-PATCH_SIZE+1):
        target = y[int(i+PATCH_SIZE/2), int(j+PATCH_SIZE/2)]
        if target == 0 :
            continue
        else :
            image_patch=Patch(X,i,j)
            #print (image_patch.shape)
            X_test_image = image_patch.reshape(1,image_patch.shape[2],image_patch.shape[0],
                                                image_patch.shape[1]).astype('float32')
            y_pred1 = (model.predict(X_test_image))
            prediction=np.argmax(y_pred1,axis=1)
            outputs[int(i+PATCH_SIZE/2)][int(j+PATCH_SIZE/2)] = prediction+1
```

Hence the model is trained, tested and validated to produce most accurate results.

CHAPTER 7: EXPERIMENTAL RESULTS

For every class in the designated 16 classes of the ground truth image that is obtained after classification, model evaluation matrices are calculated. These matrices are used to check the precision, f1-score (harmonic mean of the precision and recall), recall (the sum of true positives and false negatives) and support (the number of samples of the true response that lie in the class). The same is showed in Figure 5.

	precision	recall	f1-score	support
Alfalfa	1.00	1.00	1.00	11
Corn-notill	0.97	0.94	0.95	357
Corn-mintill	0.97	0.98	0.97	208
Corn	0.97	0.97	0.97	59
Grass-pasture	1.00	0.97	0.98	121
Grass-trees	0.99	0.99	0.99	183
Grass-pasture-mowed	1.00	1.00	1.00	7
Hay-windrowed	1.00	1.00	1.00	120
Oats	0.83	1.00	0.91	5
Soybean-notill	0.95	0.95	0.95	243
Soybean-mintill	0.96	0.98	0.97	614
Soybean-clean	0.95	0.95	0.95	148
Wheat	1.00	1.00	1.00	51
Woods	1.00	1.00	1.00	316
Buildings-Grass-Trees-Drives	0.95	0.97	0.96	97
Stone-Steel-Towers	1.00	1.00	1.00	23
accuracy			0.97	2563
macro avg	0.97	0.98	0.98	2563
weighted avg	0.97	0.97	0.97	2563

FIGURE 5: STATISTICAL ANALYSIS OF CLASSIFICATION

Output obtained after calculating precision, recall, f-score and support are used to construct a confusion matrix. The matrix hence constructed gives us information about the number of right and wrong predictions with count values and it also shows how the classification model is confused through the predictions. It gives us information about the errors made and the type of error. The confusion matrix for the classification model is employed in this study as shown in the Figure 6.

[11	0	0	0	0	0	0	0	0	0	0	0	0	0	0]
[0	337	2	1	0	0	0	0	0	3	13	1	0	0	0]
[0	1	203	0	0	0	0	0	0	0	3	1	0	0	0]
[0	1	1	57	0	0	0	0	0	0	0	0	0	0	0]
[0	0	0	0	117	0	0	0	0	0	0	1	0	0	3]
[0	0	0	0	0	182	0	0	0	0	0	0	0	0	1]
[0	0	0	0	0	0	7	0	0	0	0	0	0	0	0]
[0	0	0	0	0	0	0	120	0	0	0	0	0	0	0]
[0	0	0	0	0	0	0	0	5	0	0	0	0	0	0]
[0	3	1	0	0	0	0	0	0	230	7	2	0	0	0]
[0	4	1	1	0	0	0	0	0	7	600	0	0	0	1]
[0	3	1	0	0	0	0	0	1	2	0	141	0	0	0]
[0	0	0	0	0	0	0	0	0	0	0	0	51	0	0]
[0	0	0	0	0	0	0	0	0	0	0	0	0	316	0]
[0	0	0	0	0	1	0	0	0	0	0	2	0	0	94]
[0	0	0	0	0	0	0	0	0	0	0	0	0	0	23]]

FIGURE 6: CONFUSION MATRIX OBTAINED DURING CLASSIFICATION

✚

TABLE 2 given below summarizes the result of our classification showing the percentage of loss and the accuracy. We have observed that the Inception module of CNN model achieves greater detection performance as the average accuracy is 98.08%.

TABLE 2: ACCURACY METRICS

Metrics Accuracy	In Percentage (%)
1. Test loss	7.55130105622482
2. Test accuracy	97.30784237222005
3. Kappa accuracy	96.92939627277772
4. Overall accuracy	97.30784237222005
5. Average accuracy	98.0812132164497

Test loss: Test loss is a number indicating how bad the model's prediction was on a single example. If the model's predictions deviate too much from actual results, test loss function would cough up a very large number.

Test accuracy: Accuracy in Machine Learning is the amount of correct classifications / the total amount of classifications. The test accuracy is the accuracy of a model on examples it has not seen.

Kappa Accuracy: The Kappa statistic (or value) is a metric that compares an Observed Accuracy with an Expected Accuracy (random chance). The kappa statistic is used not only to evaluate a single classifier, but also to evaluate classifiers amongst themselves.

Overall accuracy: Overall accuracy is defined as the accuracy achieved by the model in the test set.

Average accuracy: It is the arithmetic mean of all the accuracies obtained.

The Classification map for the ground truth image (Figure 4) which is designated into 16 classes is obtained as shown in Figure 7.

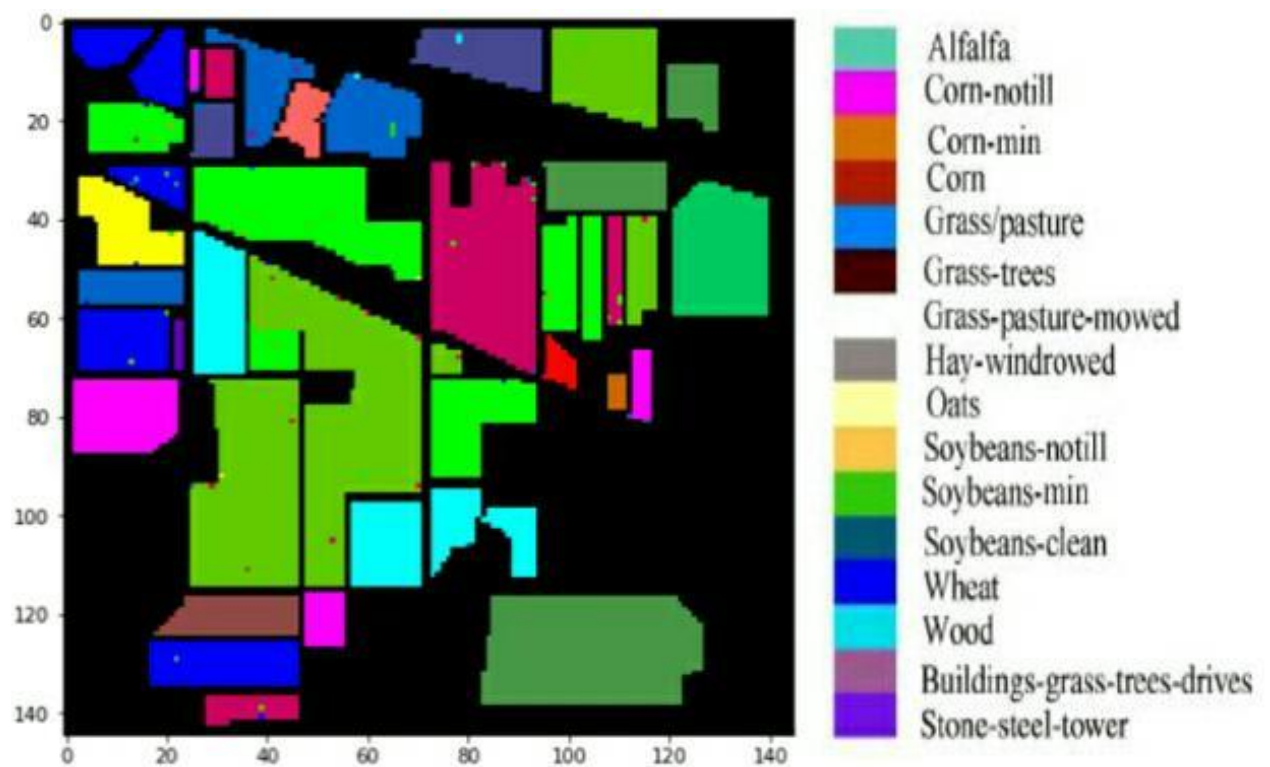


FIGURE 7: CLASSIFICATION MAP OBTAINED AFTER APPLYING CLASSIFICATION MODEL ON INDIAN PINES DATA SET

CHAPTER 8: CONCLUSION AND FUTURE SCOPE

In our paper, CNN model involving inception module is employed by using deep learning tools to implement the classification task. Inception module combines the input tensors with n number of filters and connects the obtained results to one block which results in greater performance of the classification model. This method systematically constructs high-level features that binds pixel's spectral and spatial information and presents high-level performance for every hyperspectral satellite image.

As we know PCA has done a great job in case of dimensionality reduction, there are few other methods that can be implemented to get better accuracy depending on the use case. The above method can be used in classification of satellite images of an area in particular considered for the future studies to observe the changes and different resources in that area such as agricultural land, water, built-up areas, etc.

As a part of future work, we like to gather the dataset of a region for certain time period, let's say for 10 years, on which the classification can be applied to the dataset collected from every year and the changes observed in land use/land cover patterns can be taken down. By studying these patterns obtained over years, we can make decisions on how vegetation has changed how it will help in taking measures for improving cultivation, changes in built-up areas gives information on the rate of urbanization on which several actions can be taken to control, changes in water areas will help in reducing water scarcity. This land use/land cover patterns describes the impact of human activities on the environment which plays an important role in the planning and development of the area.

CHAPTER 9: REFERENCES

- [1] Christian Szegedy, Wei Liu, YangqingJia, Pierre Sermanet, Scott Reed, DragomirAnguelov, DumitruErhan, Vincent Vanhoucke, AndrewRabinovich ,“Going Deeper With Convolutions”(IEEE)2015.
- [2] Lokman, G., & Yilmaz, G. “Hyperspectral image classification using Support Vector Neural Network algorithm”, 7th International Conference on Recent Advances in Space Technologies (RAST) 2015.
- [3] IdoFaran, Nathan S. Netanyahu, Maxim Shoshany, FadiKizel, Eli (Omid) David, JisungGeba Chang, RonitRud, “Ground Truth Simulation For Deep Learning Classification Of Mid-Resolution Venus Images Via Unmixing Of High-Resolution Hyperspectral Fenix Data”, IGARSS 2019, IEEE International Geoscience and Remote Sensing Symposium 2019.
- [4] Liu, B., Yu, X., Zhang, P., Yu, A., Fu, Q., & Wei, X, “Supervised Deep Feature Extraction for Hyperspectral Image Classification”. IEEE Transactions on Geoscience and Remote Sensing, 56(4), 1909–1921.(2018).
- [5] Amir Ghaderi, VassilisAthitsos, “Selective Unsupervised Feature Learning With Convolutional Neural Network (S-CNN)” 2016.
- [6] Lu, Y., Perez, D., Dao, M., Kwan, C., & Li, J., “Deep Learning with Synthetic Hyperspectral Images for Improved Soil Detection in Multispectral Imagery”, 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) 2018.
- [7] Shen, Y., Xiao, L., Chen, J., & Pan, D., “A Spectral-Spatial Domain-Specific Convolutional Deep Extreme Learning Machine for Supervised Hyperspectral Image Classification” , IEEE Access, 2019.
- [8]K. Makantasis, K. Karantzalos, A. Doulamis and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, 2015.

- [9] H. S. Obaid, S. A. Dheyab and S. S. Sabry, "The Impact of Data Pre-Processing Techniques and Dimensionality Reduction on the Accuracy of Machine Learning." 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), Jaipur, India, 2019.
- [10] Fan Wang, Rong Zhang and Qian Wu. "Hyperspectral Image Classification Based On PCA Network,"2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS) 2019.
- [11] Selin Bostan, Mehmet Akif Ortak, Caglayan Tuna, AlperAkoguz, ElifSertel and Burak Berk Ustundag. "Comparison of Classification Accuracy of Co-located Hyperspectral & Multispectral Images for Agricultural Purposes", Fifth International Conference on Agro-Geoinformatics 2016.
- [12] Feng Zhou, Renlong Hang, Qingshan Liu and Xiaotong Yuan. "Pyramid Fully Convolutional Network for Hyperspectral and Multispectral Image Fusion" IEEE 2019.
- [13] Hao Chen and C.H Chen, Univ of Massachusetts Dartmouth. "Hyperspectral Image Data Unsupervised Classification Using Gauss-Markov Random Fields and PCA Principle", International Geoscience and Remote Sensing Symposium IEEE 2002.
- [14] Liren Zhang, Hao Lu, Hang Zhang, Yuefeng Zhao, Huaqiang Xu and Jingjing Wang. "Hyper-Spectral Characteristics in Support of Object Classification and Verification", IEEE Access 2019.
- [15] Aamir Naveed Abbasi, Mingyi He, "Convolutional Neural Network with PCA and Batch Normalization for Hyperspectral Image Classification" IGARSS 2019, IEEE International Geoscience and Remote Sensing Symposium 2019.

APPENDICES

APPENDIX A: Registration fee paid receipt



Payment receipt of IACIT 2020

1 message

ACIT conf 2020 <acitconf@reva.edu.in>
To: smiths25398@gmail.com <smiths25398@gmail.com>

Sun, May 3, 2020 at 16:31

REVA University
School of Computing and Information Technology
Payment Receipt

Bill No:586

Date: 28/4/2020

Received an amount of Rs. 9000 (In words: Nine Thousand only)

From Mr. /Ms. SMITHA N Of REVA University towards the registration fee through online for the second International Conference on Advances in Computing and Information Technology (IACIT-2020) and publication in Scopus/Google Scholar indexed journal for the Paper titled Hyperspectral Satellite Image Classification Using Deep Learning and paper ID 586

Received by,

IACIT Team

APPENDIX B: Conference paper presentation certificate



Certificate of Presentation

School of Computing and Information Technology

This is to certify that

Mr./Mrs. SMITHA N

of REVA UNIVERSITY

has participated and presented a paper titled.....

HYPERSPECTRAL SATELLITE IMAGE

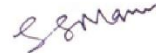
CLASSIFICATION USING DEEP LEARNING

in Second International Conference on "ADVANCES IN COMPUTING AND INFORMATION TECHNOLOGY" (IACIT-2020) held on 29th and 30th April, 2020 organized by School of Computing and Information Technology, REVA University, Bengaluru.

Co Sponsored by    ATS Learning Solutions



Dr. Mallikarjuna Shastri, P. M
General Chair, IACIT-2020
Professor, School of C&IT



Dr. Sunilkumar S Manvi
Convener, IACIT-2020
Director, School of C&IT



Dr. Surendra Rao Shankapal
Vice-Chancellor (I/C)
REVA University

Rukmini Educational
Charitable Trust

Certificate of Presentation

School of Computing and Information Technology

This is to certify that

Mr./Mrs. **SRIPRIYA M**

of **REVA UNIVERSITY**

has participated and presented a paper titled.....

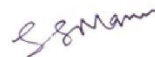
**HYPERSPECTRAL SATELLITE IMAGE
CLASSIFICATION USING DEEP LEARNING**

in Second International Conference on "ADVANCES IN
COMPUTING AND INFORMATION TECHNOLOGY"
(IACIT-2020) held on 29th and 30th April, 2020 organized
by School of Computing and Information Technology,
REVA University, Bengaluru.

Co Sponsored by    ATS Learning Solutions



Dr. Mallikarjuna Shastry, P. M
General Chair, IACIT-2020
Professor, School of C&IT



Dr. Sunil Kumar S Manvi
Convener, IACIT-2020
Director, School of C&IT



Dr. Surendra Rao Shankapal
Vice-Chancellor (I/C)
REVA University


Certificate of Presentation

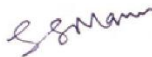
School of Computing and Information Technology


This is to certify that
Mr./Mrs. **SRIVATSA B**
of **REVA UNIVERSITY**
has participated and presented a paper titled.....
HYPER SPECTRAL SATELLITE IMAGE
CLASSIFICATION USING DEEP LEARNING

in Second International Conference on "ADVANCES IN
COMPUTING AND INFORMATION TECHNOLOGY"
(IACIT-2020) held on 29th and 30th April, 2020 organized
by School of Computing and Information Technology,
REVA University, Bengaluru.

Co Sponsored by    ATS Learning Solutions


Dr. Mallikarjuna Shastri, P. M
General Chair, IACIT-2020
Professor, School of C&IT


Dr. Sunil Kumar S Manvi
Convener, IACIT-2020
Director, School of C&IT


Dr. Surendra Rao Shankapal
Vice-Chancellor (I/C)
REVA University

Certificate of Presentation

School of Computing and Information Technology

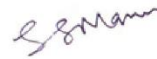
This is to certify that
Mr./Mrs. **SUMA MV**
of **REVA UNIVERSITY**
has participated and presented a paper titled.....
***HYPERSPECTRAL SATELLITE IMAGE
CLASSIFICATION USING DEEP LEARNING***

in Second International Conference on "ADVANCES IN
COMPUTING AND INFORMATION TECHNOLOGY"
(IACIT-2020) held on 29th and 30th April, 2020 organized
by School of Computing and Information Technology,
REVA University, Bengaluru.

Co Sponsored by    ATS Learning Solutions



Dr. Mallikarjuna Shastry, P. M
General Chair, IACIT-2020
Professor, School of C&IT



Dr. Sunil Kumar S Manvi
Convener, IACIT-2020
Director, School of C&IT



Dr. Surendra Rao Shankapal
Vice-Chancellor (I/C)
REVA University

Certificate of Presentation

School of Computing and Information Technology

This is to certify that

Mr./Mrs. **DR. MALLIKARJUN M KODABAGI**
of **REVA UNIVERSITY**

has participated and presented a paper titled.....
***HYPERSPECTRAL SATELLITE IMAGE
CLASSIFICATION USING DEEP LEARNING***

in Second International Conference on "ADVANCES IN
COMPUTING AND INFORMATION TECHNOLOGY"
(IACIT-2020) held on 29th and 30th April, 2020 organized
by School of Computing and Information Technology,
REVA University, Bengaluru.

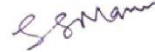
Co Sponsored by



ATS Learning Solutions



Dr. Mallikarjuna Shastry, P. M
General Chair, IACIT-2020
Professor, School of C&IT



Dr. Sunil Kumar S Manvi
Convener, IACIT-2020
Director, School of C&IT



Dr. Surendra Rao Shankapal
Vice-Chancellor (I/C)
REVA University

APPENDIX C: Paper

Hyperspectral Satellite Image Classification Using Deep Learning

SMITHA N, SRIPRIYA M, SRIVATSA B, SUMA MV, DR. MALLIKARJUN M
KODABAGI

School of Computing and Information Technology, REVA University, Bangalore, India

Email: smiths25398@gmail.com, priyamadhan@yahoo.com, svtsbalaji.208@gmail.com,
sumavenkat98@gmail.com, mallikarjunmk@reva.edu.in

Abstract—Hyperspectral images are utilized to provide adequate spectral information so as to acknowledge and differentiate spectrally distinctive materials. Optical analysis techniques are utilized to detect and identify the objects from a scale of images. Hyperspectral imaging technique is one among them. Hyperspectral image classification research is an intense field of study and an outsized number of recent approaches have been developed to enhance the performance for specific applications that exploit both spatial and spectral image content. The goal of hyperspectral imaging is to obtain the spectrum for each and every pixel within the image of a scene, with the intent of detecting processes, identifying materials or finding objects. In this particular study, a strategy for the classification of Hyperspectral satellite images is asserted using deep learning framework. This framework involves inception module architecture containing 1x1, 3x3 and 5x5 Convolutional layers which gives an overall classification accuracy of 97.30%.

Keywords - Hyperspectral, classification, inception module

I. INTRODUCTION

Remote sensing methods are any methods where information/data is extracted or interpreted by indirect measurement of the thing under study (I.e. there's no physical contact with the object). The data is obtained via non-particulate radiation.

Remote sensing allows coverage of very large areas which enables regional surveys on a spread of themes and identification of extremely large features. It allows repeated coverage which proves

to be very useful when accumulating data on varied dynamic themes like water, agricultural fields and so on.

Hyperspectral images are volumetric image cubes that encompass many spatial images. Each spatial image, or spectral band, captures the responses of ground objects at a selected wave length. Due to its rich spatial and spectral information contents, hyperspectral imagery has become a vital tool for a spread of remote sensing and scanning applications. It's widely employed in the sensing and discovering of ground minerals, in monitoring of the Earth's resources, and in military surveillance.

In the approach discussed here an Inception module [1] is used in order to perform the classification. Using this model we can extract features and classify Indian pines hyperspectral images more efficiently and easily.

II. LITERATURE SURVEY

Lokman G. & Yilmaz, G. in their paper on hyperspectral image classification using SVM [2] have designed the classifier using the vector neural network for Hyperspectral image which the neural network is trained for Eigen-value decay. The pre-processed hyperspectral image data is used as the input to the system. It contains classification of the image in matrix form as the output of the system. The output of the target class is in negative values and the background data with other values. To test the classifier they used Hyperion sensor on NASA EO-1 satellite to classify the airborne visible/infrared imaging spectrometer (AVIRIS) image of Okavango Delta and image of Salinas Valley.

In this paper [3], IdoFaran, Nathan S. Netanyahu, Maxim Shoshany, FadiKizel, Eli (Omid) David, Jisung Geba Chang, RonitRud have proposed a novel method, to overcome the problem of lack of adequate data by training a deep neural network for classification. For training the convolutional neural network pixel based classification, they used simulated ground truth from hyperspectral, high resolution FENIX image.

[4] In this paper to improve the performance of the hyperspectral image classification based on Siamese convolution neural network (S-CNN) using supervised deep feature extraction method they use labelled samples for training. It is separated into different classes by training the S-CNN [5]. The feature of hyperspectral cube is extracted from five layer Convolution neural network of nonlinear transformation function. The classification task is extracted discriminative by using supervised margin ranking loss function. Thus the support vector machine provides the better classification performance than the conventional method

In this paper [6], Lu, Y., Perez, D., Dao, M., Kwan, C., & Li, J. have collected the image from WorldView-2 satellite for soil detection by combination of original 8 multispectral bands and 80 synthetic hyperspectral bands. The CNN model is used for soil detection over the curved surface area has increased its average of 7.42% in average from 76.26 to 83.48. Soil detection performance can also be increased by pan-sharpening and morphological post processing. For improving the performance of classifying the remote sensing application and object detection can perform synthetic hyperspectral bands of CNN model.

In this paper[7], Shen, Y., Xiao, L., Chen, J., & Pan, D. have used for novel deep ELM neural network for Indian Pines dataset, Pavia University dataset and Salinas dataset for testing. The fully connected deep ELM network and two branch convolutional learning module within hidden nodes are the two part of framework. The generating random weight to Hyperspectral image spectral and spatial features are extracted, concentrated and fed into the fully connected ELM networks. The classified result is obtained from ELM network.

In this paper [8], AVIRIS and ROSIS hyperspectral datasets of developed framework of deep learning are used for experimentation and validation. The

spatial and spectral information are encoded using the high level features of CNNs and MLPs. They compared the performance of SVM based classifier and deep learning approach. For every data set given for the validation, deep learning approach has the superior performance.

III. METHODOLOGY

In this approach (Figure 1) to classify the satellite image data hyperspectral satellite images (HSI) are provided as input. These hyperspectral images are composed of multiple bands/channels which contain spectral and special information about the pixels. By considering a single hyperspectral image (Figure 2) and its ground truth image (Figure 3) as input data, classification is performed to classify each pixel into its corresponding class. Ground truth data gives information about the different predefined classes of the pixels.

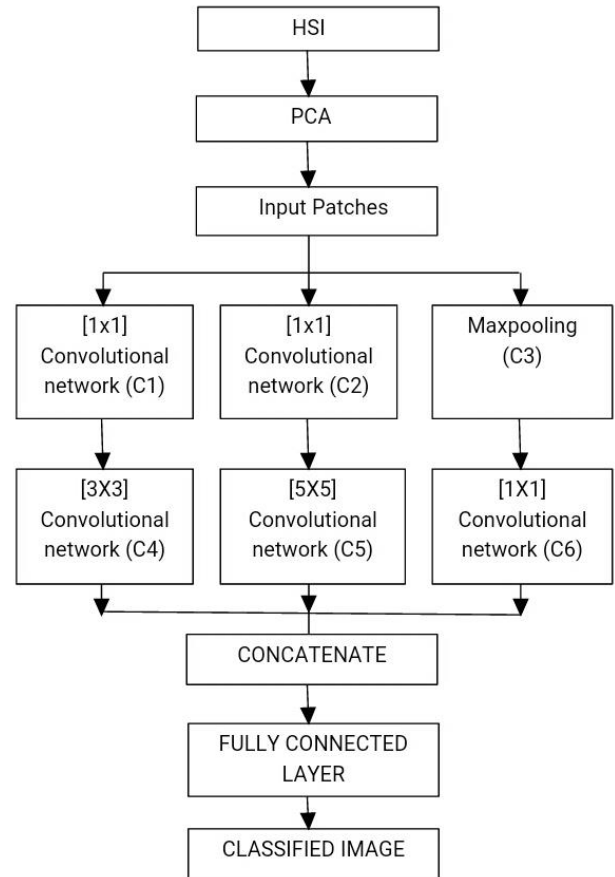


FIGURE 3: BLOCK DIAGRAM OF OVERALL SYSTEM ARCHITECTURE

The hyperspectral image is represented as a 3 Dimensional tensor of $W \times H \times B$ dimension, where W is the width, H is the height of the image and B denotes the number of spectral bands/channels consisting of several spectral and spatial information.



FIGURE 2: HYPERSPECTRAL IMAGE

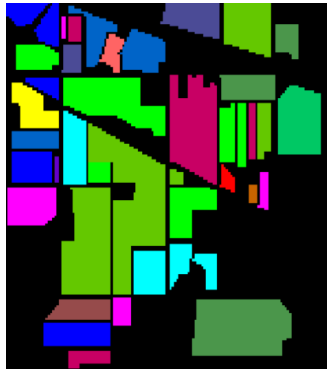


FIGURE 3: GROUNDTRUTH IMAGE

A. Dimensionality reduction of the input image:

The input image containing hundreds of spectral bands lead to high dimensionality which may result in more training time and high computational costs. From an analysis made on spectral response patterns of pixels, it is observed that the pixels which belong to same class have a very small variance in their response patterns. This predicts that these pixels have similar properties and almost the same pixel values in every channel, while the pixels belonging to different classes have different properties due to large amount of variance. Taking this into account dimensionality reduction is achieved by employing PCA (Principal Component Analysis) [9] approach along the spectral dimension to abridge the entire image into lower dimension. PCA reduces few spectral properties

which possess less variance by preserving the spatial properties.

In this experimentation process the first 30 principle components are retained to preserve most of the initial information. After the application of PCA, the input image of dimension $W \times H \times B$ is reduced to $W \times H \times B_r$, where B_r is the number of bands after the application of PCA.

B. Creation of Input patches:

The hyperspectral image with reduced dimensions after applying PCA is further decomposed into numerous patches of dimension $S \times S \times B_r$ which are also 3 Dimensional tensors. Here S corresponds to length of the square patch. To classify a pixel at location (x, y) to the corresponding class label $L(x, y)$, a patch $P(x, y)$ of dimension $S \times S \times B_r$ centred at that pixel is used. Hence the dataset contains pixel's information in the form $D = \{P(x, y), L(x, y)\}$ where $x = 1, 2, \dots, W$ and $y = 1, 2, \dots, H$.

S is kept minimum as 5 in order to have the neighboring pixels as minimum as possible during the classification process which may otherwise increase the computational cost of training data.

C. Classification model:

Input patches obtained after applying PCA with reduced dimensions are sent to the deep learning classification structure. 25% of the dataset is used for testing and remaining data is used to train the classification model. The model has to train all kinds of samples with maximum accuracies. The Convolutional Neural Network (CNN) architecture which is used here is similar to inception module. It consists of convolutional layers $C1$, $C2$ with trainable filters of dimension 1×1 and $C3$ as maxpooling layer. The $C1$ and $C2$ convolutional layers are followed by a second set of convolutional layers $C4$ and $C5$ with 60 trainable filters, the filters are of dimension 3×3 and 5×5 respectively. By adding 1×1 Convolutional layer before the 3×3 and 5×5 Convolutional layers, while keeping the height and width of the feature map, the number of computations are reduced by a factor of 10. This reduces the computational requirements and in turn maximum efficiency is also obtained.

C3 which is maxpooling layer is followed by convolution layer C6 with dimension 1x1. The output from each of C4, C5 and C6 layers is concatenated to form output of the inception module. The output obtained is then flattened into one dimensional set of neurons which is further used to create a fully connected neural network layer for final classification. The model was trained for 25 epochs. The training accuracy of the model increased to a maximum in every epoch. A classified image with each pixel classified into its respective predefined class labels is obtained as the final output.

IV. EXPERIMENTAL RESULTS

A. Dataset:

In this experiment the Indian Pines data set gathered by the AVIRIS sensor over the Indian Pines test site in North-western Indiana is used. It consists of 224 spectral reflectance bands within wavelength ranging between 0.4 and 2.5 x 10⁻⁶ meters and 145x145 pixels. Here 75% of data set sample is randomly selected as training dataset and remaining 25% as testing dataset. The Indian Pines data consists of two-third of agriculture and one-third of forest/other natural vegetation. The ground truth image of the dataset is designated into 16 classes. Each class consists of various amount of samples. The total number of samples present in the dataset is 10249. Once the classification is done, the results are obtained as discussed below.

B. Result:

For each class in the designated sixteen classes of the ground truth image which is obtained after classification, the model evaluation matrices are calculated. These evaluation matrices are used to check the precision, recall (the sum of true positives and false negatives), f1-score (harmonic mean of the precision and recall) and support (the number of samples of the true response that lie in the class). The same is showed in Figure 4.

Values obtained by calculating precision, recall, f-score and support are used to build a confusion matrix. This matrix gives information about the number of correct and incorrect predictions with count values and also shows how our classification

	precision	recall	f1-score	support
Alfalfa	1.00	1.00	1.00	11
Corn-notill	0.97	0.94	0.95	357
Corn-mintill	0.97	0.98	0.97	208
Corn	0.97	0.97	0.97	59
Grass-pasture	1.00	0.97	0.98	121
Grass-trees	0.99	0.99	0.99	183
Grass-pasture-mowed	1.00	1.00	1.00	7
Hay-windrowed	1.00	1.00	1.00	120
Oats	0.83	1.00	0.91	5
Soybean-notill	0.95	0.95	0.95	243
Soybean-mintill	0.96	0.98	0.97	614
Soybean-clean	0.95	0.95	0.95	148
Wheat	1.00	1.00	1.00	51
Woods	1.00	1.00	1.00	316
Buildings-Grass-Trees-Drives	0.95	0.97	0.96	97
Stone-Steel-Towers	1.00	1.00	1.00	23
accuracy			0.97	2563
macro avg	0.97	0.98	0.98	2563
weighted avg	0.97	0.97	0.97	2563

FIGURE 4: STATISTICAL ANALYSIS OF CLASSIFICATION

model is confused during predictions. It also gives us information about the errors being made and their types. The confusion matrix obtained for the classification model employed in this study is shown in the Figure 5.

[11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0]
[0	337	2	1	0	0	0	0	0	3	13	1	0	0	0	0]
[0	1	203	0	0	0	0	0	0	0	3	1	0	0	0	0]
[0	1	1	57	0	0	0	0	0	0	0	0	0	0	0	0]
[0	0	0	0	117	0	0	0	0	0	0	1	0	0	3	0]
[0	0	0	0	0	182	0	0	0	0	0	0	0	0	1	0]
[0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0]
[0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	0]
[0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0]
[0	3	1	0	0	0	0	0	0	230	7	2	0	0	0	0]
[0	4	1	1	0	0	0	0	0	7	600	0	0	0	1	0]
[0	3	1	0	0	0	0	0	1	2	0	141	0	0	0	0]
[0	0	0	0	0	0	0	0	0	0	0	0	51	0	0	0]
[0	0	0	0	0	0	0	0	0	0	0	0	0	316	0	0]
[0	0	0	0	0	1	0	0	0	0	0	2	0	0	94	0]
[0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23]]

FIGURE 5: CONFUSION MATRIX OBTAINED DURING CLASSIFICATION

TABLE - I summarizes the result of the classification showing the loss percentage and the accuracy percentage. It is observed that the Inception module of CNN model achieves significant detection performance as the average accuracy is 98.08%.

TABLE - I

Metrics Accuracy	In Percentage (%)
6. Test loss	7.55130105622482
7. Test accuracy	97.30784237222005
8. Kappa accuracy	96.92939627277772
9. Overall accuracy	97.30784237222005
10. Average accuracy	98.0812132164497

The Classification map for the ground truth image (Figure 3) which is designated into 16 classes is obtained as shown in Figure 6.

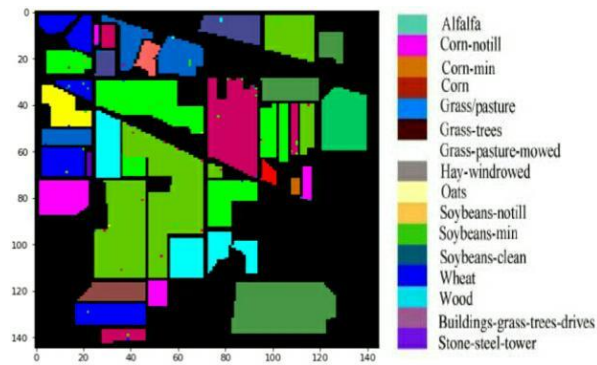


FIGURE 6: CLASSIFICATION MAP OBTAINED AFTER APPLYING CLASSIFICATION MODEL ON INDIAN PINES DATA SET

V. CONCLUSION AND FUTURE SCOPE

In this paper, an Inception module architecture of CNN model using deep learning tools is implemented for the classification task. Inception module convolves the input tensors with multiple number of filters and connects the obtained results into one block which results in higher performance of the classification model. This approach systematically constructs superior features that encodes pixel's spectral and spatial details and presents high-level performance for every hyperspectral satellite image data.

Although PCA has done a good job in case of dimensionality reduction, there are other methods which can be implemented to yield better accuracy depending upon the use case. The above approach can be used in performing classification of satellite images of a particular area considered for study to

observe the different resources in that area such as agricultural land, water, built-up areas, etc.

As a part of future work, we would like to gather the dataset of a particular region for a particular time period, say 10 years, on which the discussed classification approach can be applied to the dataset collected for every year and the changes in land use/land cover patterns can be detected. By studying these patterns, observations can be made on following factors such as vegetation changes which will help in taking measures for improving cultivation patterns, change in built-up areas which denotes the rate of urbanization on which several actions can be taken to control the rate, changes in water covered areas which will help in reducing water scarcity. This land use/land cover pattern describes the impact of human activities on the environment which plays a major role in the planning and development of a region.

REFERENCES

- [1] Christian Szegedy, Wei Liu, YangqingJia, Pierre Sermanet, Scott Reed, DragomirAnguelov, DumitruErhan, Vincent Vanhoucke, AndrewRabinovich, "Going Deeper With Convolutions"(IEEE)2015.
- [2] Lokman, G., & Yilmaz, G. "Hyperspectral image classification using Support Vector Neural Network algorithm", 7th International Conference on Recent Advances in Space Technologies (RAST) 2015.
- [3] IdoFaran, Nathan S. Netanyahu, Maxim Shoshany, FadiKizel, Eli (Omid) David, Jisung Geba Chang, RonitRud, "Ground Truth Simulation For Deep Learning Classification Of Mid-Resolution Venus Images Via Unmixing Of High-Resolution Hyperspectral Fenix Data", IGARSS 2019, IEEE International Geoscience and Remote Sensing Symposium 2019.
- [4] Liu, B., Yu, X., Zhang, P., Yu, A., Fu, Q., & Wei, X, "Supervised Deep Feature Extraction for Hyperspectral Image Classification". IEEE Transactions on Geoscience and Remote Sensing, 56(4), 1909–1921.(2018).
- [5] Amir Ghaderi, Vassilis Athitsos, "Selective Unsupervised Feature Learning With Convolutional Neural Network (S-CNN)" 2016.
- [6] Lu, Y., Perez, D., Dao, M., Kwan, C., & Li, J., "Deep Learning with Synthetic Hyperspectral Images for Improved Soil Detection in

Multispectral Imagery”, 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) 2018.

[7] Shen, Y., Xiao, L., Chen, J., & Pan, D., “A Spectral-Spatial Domain-Specific Convolutional Deep Extreme Learning Machine for Supervised Hyperspectral Image Classification” , IEEE Access, 2019.

[8]K. Makantasis, K. Karantzas, A. Doulamis and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, 2015.

[9] H. S. Obaid, S. A. Dheyab and S. S. Sabry, "The Impact of Data Pre-Processing Techniques and Dimensionality Reduction on the Accuracy of Machine Learning." 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), Jaipur, India, 2019.

[10] Fan Wang, Rong Zhang and Qian Wu. “Hyperspectral Image Classification Based On PCA Network,”2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS) 2019.

[11] Selin Bostan, Mehmet Akif Ortak, Caglayan Tuna, Alper Akoguz, Elif Sertel and Burak Berk Ustundag. “Comparison of Classification Accuracy of Co-located Hyperspectral & Multispectral Images for Agricultural Purposes”, Fifth International Conference on Agro-Geoinformatics 2016.

[12] Feng Zhou, Renlong Hang, Qingshan Liu and Xiaotong Yuan. “Pyramid Fully Convolutional Network for Hyperspectral and Multispectral Image Fusion” IEEE 2019.

[13] Hao Chen and C.H Chen, Univ of Massachusetts Dartmouth. “Hyperspectral Image Data Unsupervised Classification Using Gauss-Markov Random Fields and PCA Principle”, International Geoscience and Remote Sensing Symposium IEEE 2002.

[14] Liren Zhang, Hao Lu, Hang Zhang, Yuefeng Zhao, Huaqiang Xu and Jingjing Wang. “Hyper-Spectral Characteristics in Support of Object Classification and Verification”, IEEE Access 2019.

[15] Aamir Naveed Abbasi, Mingyi He, “Convolutional Neural Network with PCA and

Batch Normalization for Hyperspectral Image Classification” IGARSS 2019, IEEE International Geoscience and Remote Sensing Symposium 2019.

AUTHORS PROFILE

Smitha N is pursuing B.Tech (Computer Science and Engineering) in REVA University, Bangalore. Her subjects of interest include Machine Learning, Artificial Intelligence and Data Mining.

Sri Priya M is pursuing B.Tech (Computer Science and Engineering) in REVA University, Bangalore. Her subjects of interests are Image Processing, Pattern Recognition and Machine Learning.

Srivatsa B is pursuing B.Tech (Computer Science and Engineering) in REVA University, Bangalore. His subjects of interests are Data Science, Machine Learning, App Development and Web Development.

Suma MV is pursuing B.Tech (Computer Science and Engineering) in REVA University, Bangalore. Her subjects of interests are Machine Learning, Neural Networks and Soft Computing.

Dr.Mallikarjun M Kodabagi is a Professor at School of C&IT, REVA University, Bangalore.

APPENDIX D: Plagiarism check certificate

PID-586

by Smitha N

Submission date: 17-Apr-2020 06:36PM (UTC+0530)

Submission ID: 1300013987

File name: 586-IACIT-20-586_Revised_version.docx (1.41M)

Word count: 2611

Character count: 14973

PID-586

ORIGINALITY REPORT

20%

SIMILARITY INDEX

11%

INTERNET SOURCES

14%

PUBLICATIONS

9%

STUDENT PAPERS

PRIMARY SOURCES

- | | | |
|---|---|----|
| 1 | Ngai-Man Cheung, Antonio Ortega. "Distributed Compression of Hyperspectral Imagery", Elsevier BV, 2009
Publication | 3% |
| 2 | www.ngi.gov.za
Internet Source | 2% |
| 3 | Muhammad Mateen, Junhao Wen, Nasrullah -, Muhammad Azeem. "The Role of Hyperspectral Imaging: A Literature Review", International Journal of Advanced Computer Science and Applications, 2018
Publication | 2% |
| 4 | "Advances in Visual Computing", Springer Science and Business Media LLC, 2015
Publication | 1% |
| 5 | Yan Lu, Daniel Perez, Minh Dao, Chiman Kwan, Jiang Li. "Deep Learning with Synthetic Hyperspectral Images for Improved Soil Detection in Multispectral Imagery", 2018 9th IEEE Annual Ubiquitous Computing, Electronics | 1% |

& Mobile Communication Conference
(UEMCON), 2018

Publication

-
- | | | |
|----------|---|------------|
| 6 | Submitted to Sri Lanka Institute of Information Technology
<small>Student Paper</small> | 1 % |
|----------|---|------------|
-
- | | | |
|----------|---|------------|
| 7 | Submitted to Kingston University
<small>Student Paper</small> | 1 % |
|----------|---|------------|
-
- | | | |
|----------|--|------------|
| 8 | dias.library.tuc.gr
<small>Internet Source</small> | 1 % |
|----------|--|------------|
-
- | | | |
|----------|--|------------|
| 9 | Ido Faran, Nathan S. Netanyahu, Eli Omid David, Maxim Shoshany, Fadi Kizel, Jisung Geba Chang, Ronit Rud. "Ground Truth Simulation for Deep Learning Classification of Mid-Resolution Venus Images Via Unmixing of High-Resolution Hyperspectral Fenix Data", IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium, 2019
<small>Publication</small> | 1 % |
|----------|--|------------|
-
- | | | |
|-----------|---|------------|
| 10 | H. Ma, W. Feng, X. Cao, L. Wang. "CLASSIFICATION OF HYPERSPECTRAL DATA BASED ON GUIDED FILTERING AND RANDOM FOREST", ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2017
<small>Publication</small> | 1 % |
|-----------|---|------------|
-

11	link.springer.com Internet Source	1%
12	Submitted to Embry Riddle Aeronautical University Student Paper	1%
13	Submitted to University of Southampton Student Paper	1%
14	Lokman, Gurcan, and Guray Yilmaz. "Hyperspectral image classification using Support Vector Neural Network algorithm", 2015 7th International Conference on Recent Advances in Space Technologies (RAST), 2015. Publication	1%
15	Submitted to University of Macau Student Paper	<1%
16	repository.tudelft.nl Internet Source	<1%
17	Mengying Jiang, Faxian Cao, Yunmeng Lu. "Extreme Learning Machine With Enhanced Composite Feature for Spectral-Spatial Hyperspectral Image Classification", IEEE Access, 2018 Publication	<1%
18	"Computer Vision", Springer Science and Business Media LLC, 2017 Publication	<1%

19	res.mdpi.com Internet Source	<1%
20	Mahesh Pal. "KERNEL METHODS IN REMOTE SENSING: A REVIEW", ISH Journal of Hydraulic Engineering, 2009 Publication	<1%
21	grindgis.com Internet Source	<1%
22	ermt.net Internet Source	<1%

Exclude quotes On
Exclude bibliography On

Exclude matches < 10 words

Hyperspectral Satellite Image Classification using Deep Learning

¹Smitha N, ²Sripriya M, ³Srivatsa B, ⁴Suma MV, ⁵Mallikarjun M Kodabagi

^{1,2,3,4,5}School of Computing and Information Technology, REVA University, Bangalore, India

¹smiths25398@gmail.com, ²priyamadhan@yahoo.com, ³srvtbalaji.208@gmail.com,

⁴sumavenkat98@gmail.com, ⁵mallikarjunmk@reva.edu.in

Article Info

Volume 83

Page Number: 4458-4462

Publication Issue:

May - June 2020

Abstract

Hyperspectral images are utilized to provide adequate spectral information so as to acknowledge and differentiate spectrally distinctive materials. Optical analysis techniques are utilized to detect and identify the objects from a scale of images. Hyperspectral imaging technique is one among them. Hyperspectral image classification research is an intense field of study and an outsized number of recent approaches have been developed to enhance the performance for specific applications that exploit both spatial and spectral image content. The goal of hyperspectral imaging is to obtain the spectrum for each and every pixel within the image of a scene, with the intent of detecting processes, identifying materials or finding objects. In this particular study, a strategy for the classification of Hyperspectral satellite images is asserted using deep learning framework. This framework involves inception module architecture containing 1x1, 3x3 and 5x5 Convolutional layers which gives an overall classification accuracy of 97.30%.

Article History

Article Received: 19 November 2019

Revised: 27 January 2020

Accepted: 24 February 2020

Publication: 12 May 2020

Keywords: Hyperspectral imaging technique, convolutional layers.

1. Introduction

Remote sensing methods are any methods where information/data is extracted or interpreted by indirect measurement of the thing under study (I.e. there's no physical contact with the object). The data is obtained via non-particulate radiation.

Remote sensing allows coverage of very large areas which enables regional surveys on a spread of themes and identification of extremely large features. It allows repeated coverage which proves to be very useful when accumulating data on varied dynamic themes like water, agricultural fields and so on.

Hyperspectral images are volumetric image cubes that encompass many spatial images. Each spatial image, or spectral band, captures the responses of ground objects at a selected wave length. Due to its rich spatial and spectral information contents, hyperspectral imagery has become a vital tool for a spread of remote sensing and scanning applications. It's widely employed in the sensing

and discovering of ground minerals, in monitoring of the Earth's resources, and in military surveillance.

In the approach discussed here an Inception module [1] is used in order to perform the classification. Using this model we can extract features and classify Indian pines hyperspectral images more efficiently and easily.

2. Literature Survey

Lokman G. & Yilmaz, G. in their paper on hyperspectral image classification using SVM [2] have designed the classifier using the vector neural network for Hyperspectral image which the neural network is trained for Eigen-value decay. The pre-processed hyperspectral image data is used as the input to the system. It contains classification of the image in matrix form as the output of the system. The output of the target class is in negative values and the background data with other values. To test the classifier they used Hyperion sensor on NASA EO-1 satellite to classify the airborne visible/infrared imaging

spectrometer (AVIRIS) image of Okavango Delta and image of Salinas Valley.

In this paper [3], Ido Faran, Nathan S. Netanyahu, Maxim Shoshany, Fadi Kizel, Eli (Omid) David, Jisung Geba Chang, Ronit Rud have proposed a novel method, to overcome the problem of lack of adequate data by training a deep neural network for classification. For training the convolutional neural network pixel based classification, they used simulated ground truth from hyperspectral, high resolution FENIX image.

[4] In this paper to improve the performance of the hyperspectral image classification based on Siamese convolution neural network (S-CNN) using supervised deep feature extraction method they use labelled samples for training. It is separated into different classes by training the S-CNN [5]. The feature of hyperspectral cube is extracted from five layer Convolution neural network of nonlinear transformation function. The classification task is extracted discriminative by using supervised margin ranking loss function. Thus the support vector machine provides the better classification performance than the conventional method

In this paper [6], Lu, Y., Perez, D., Dao, M., Kwan, C., & Li, J. have collected the image from WorldView-2 satellite for soil detection by combination of original 8 multispectral bands and 80 synthetic hyperspectral bands. The CNN model is used for soil detection over the curved surface area has increased its average of 7.42% in average from 76.26 to 83.48. Soil detection performance can also be increased by pan-sharpening and morphological post processing. For improving the performance of classifying the remote sensing application and object detection can perform synthetic hyperspectral bands of CNN model.

In this paper[7], Shen, Y., Xiao, L., Chen, J., & Pan, D. have used for novel deep ELM neural network for Indian Pines dataset, Pavia University dataset and Salinas dataset for testing. The fully connected deep ELM network and two branch convolutional learning module within hidden nodes are the two part of framework. The generating random weight to Hyperspectral image spectral and spatial features are extracted, concentrated and fed into the fully connected ELM networks. The classified result is obtained from ELM network.

In this paper [8], AVIRIS and ROSIS hyperspectral datasets of developed framework of deep learning are used for experimentation and validation. The spatial and spectral information are encoded using the high level features of CNNs and MLPs. They compared the performance of SVM based classifier and deep learning approach. For every data set given for the validation, deep learning approach has the superior performance.

3. Methodology

In this approach (Figure1) to classify the satellite image data hyperspectral satellite images (HSI) are provided as input. These hyperspectral images are composed of multiple bands/channels which contain spectral and

special information about the pixels. By considering a single hyperspectral image (Figure2) and its ground truth image (Figure 3) as input data, classification is performed to classify each pixel into its corresponding class. Ground truth data gives information about the different predefined classes of the pixels.

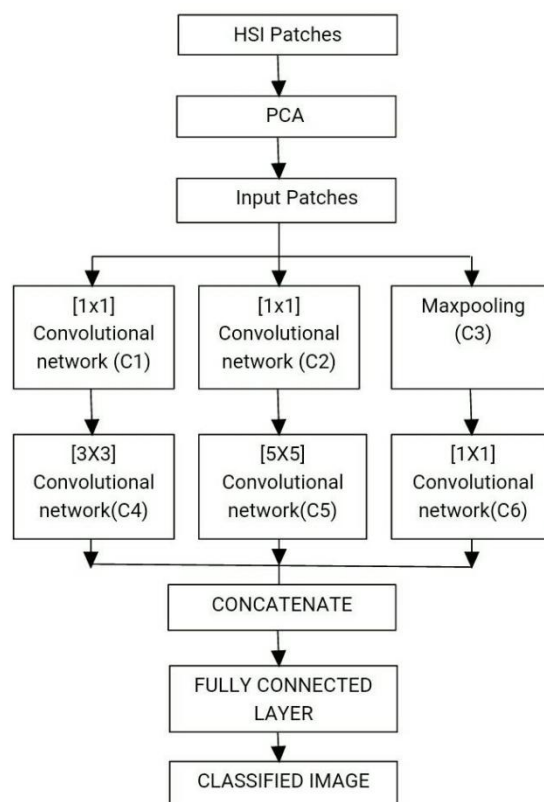


Figure 1: Block diagram of overall system architecture

The hyperspectral image is represented as a 3 Dimensional tensor of $W \times H \times B$ dimension, where W is the width, H is the height of the image and B denotes the number of spectral bands/channels consisting of several spectral and spatial information. This image is further decomposed into numerous patches of dimension $S \times S \times B$. To classify a pixel at location (x, y) to the corresponding class label $L(x, y)$, a patch $P(x, y)$ of dimension $S \times S \times B$ centred at that pixel is used. Hence the dataset contains pixel's information in the form $\{P(x, y), L(x, y)\}$ where $x=1,2,\dots,W$ and $y=1,2,\dots,H$



Figure 2: Hyperspectral Image

Dimensionality reduction of the input image:

The input patches containing hundreds of spectral bands lead to high dimensionality which may result in more training time and high computational costs. From an analysis made on spectral response patterns of pixels, it is observed that the pixels which belong to same class have a very small variance in their response patterns. This predicts that these pixels have similar properties and almost the same pixel values in every channel, while the pixels belonging to different classes have different properties due to large amount of variance. Taking this into account dimensionality reduction is achieved by employing PCA (Principal Component Analysis) [9] approach along the spectral dimension to abridge the entire image into lower dimension. PCA reduces few spectral properties which possess less variance by preserving the spatial properties.

In this experimentation process the first 30 principle components are retained to preserve most of the initial information. After the application of PCA, the input patch which is also a 3D tensor of dimension $S \times S \times B$ is reduced to $S \times S \times Br$, where Br is the number of bands after the application of PCA and S is kept minimum as 5 to have the neighbouring pixels as minimum as possible during classification process which else may increase the computational cost of training data.



Figure 3: Ground Truth Image

Classification model:

Input patches obtained after applying PCA with reduced dimensions are sent to the deep learning classification structure. 25% of the dataset is used for testing and remaining data is used to train the classification model. The model has to train all kinds of samples with maximum accuracies. The Convolutional Neural Network (CNN) architecture which is used here is similar to inception module. It consists of convolutional layers C1, C2 with trainable filters of dimension 1×1 and C3 as maxpooling layer. The C1 and C2 convolutional layers are followed by a second set of convolutional layers C4 and C5 with 60 trainable filters, the filters are of dimension 3×3 and 5×5 respectively. By adding 1×1 Convolutional layer before the 3×3 and 5×5 Convolutional layers, while keeping the height and width of the feature map, the number of computations are

reduced by a factor of 10. This reduces the computational requirements and in turn maximum efficiency is also obtained. C3 which is maxpooling layer is followed by convolution layer C6 with dimension 1×1 . The output from each of C4, C5 and C6 layers is concatenated to form output of the inception module. The output obtained is then flattened into one dimensional set of neurons which is further used to create a fully connected neural network layer for final classification. The model was trained for 25 epochs. The training accuracy of the model increased to a maximum in every epoch. A classified image with each pixel classified into its respective predefined class labels is obtained as the final output.

4. Experimental Results

Dataset:

In this experiment the Indian Pines data set gathered by the AVIRIS sensor over the Indian Pines test site in North-western Indiana is used. It consists of 224 spectral reflectance bands within wavelength ranging between 0.4 and 2.5×10^{-6} meters and 145×145 pixels. Here 75% of data set sample is randomly selected as training dataset and remaining 25% as testing dataset. The Indian Pines data consists of two-third of agriculture and one-third of forest/other natural vegetation. The ground truth image of the dataset is designated into 16 classes. Once the classification is done, the results are obtained as discussed below.

Result:

For each class in the designated sixteen classes of the ground truth image which is obtained after classification, the model evaluation matrices are calculated. These evaluation matrices are used to check the precision, recall (the sum of true positives and false negatives), f1-score (harmonic mean of the precision and recall) and support (the number of samples of the true response that lie in the class). The same is showed in Figure 4.

	precision	recall	f1-score	support
Alfalfa	1.00	1.00	1.00	11
Corn-notill	0.97	0.94	0.95	357
Corn-mintill	0.97	0.98	0.97	208
Corn	0.97	0.97	0.97	59
Grass-pasture	1.00	0.97	0.98	121
Grass-trees	0.99	0.99	0.99	183
Grass-pasture-mowed	1.00	1.00	1.00	7
Hay-windrowed	1.00	1.00	1.00	120
Oats	0.83	1.00	0.91	5
Soybean-notill	0.95	0.95	0.95	243
Soybean-mintill	0.96	0.98	0.97	614
Soybean-clean	0.95	0.95	0.95	148
Wheat	1.00	1.00	1.00	51
Woods	1.00	1.00	1.00	316
Buildings-Grass-Trees-Drives	0.95	0.97	0.96	97
Stone-Steel-Towers	1.00	1.00	1.00	23
accuracy			0.97	2563
macro avg	0.97	0.98	0.98	2563
weighted avg	0.97	0.97	0.97	2563

Figure 4: Statistical Analysis of Classification

Values obtained by calculating precision, recall, f-score and support are used to build a confusion matrix.

This matrix gives information about the number of correct and incorrect predictions with count values and also shows how our classification model is confused during predictions. It also gives us information about the errors being made and their types. The confusion matrix obtained for the classification model employed in this study is shown in the Figure 5.

```
[[ 11  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0 337  2  1  0  0  0  0  0  3 13  1  0  0  0  0]
 [ 0  1 203  0  0  0  0  0  0  0  3  1  0  0  0  0]
 [ 0  1  1 57  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0 117  0  0  0  0  0  0  1  0  0  3  0]
 [ 0  0  0  0  0 182  0  0  0  0  0  0  0  0  1  0]
 [ 0  0  0  0  0  0 7  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 120  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0 5  0  0  0  0  0  0  0]
 [ 0  3  1  0  0  0  0  0  0 230  7  2  0  0  0  0]
 [ 0  4  1  1  0  0  0  0  0  7 600  0  0  0  1  0]
 [ 0  3  1  0  0  0  0  0  1  2  0 141  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0 51  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0 316  0  0]
 [ 0  0  0  0  0  1  0  0  0  0  0  2  0  0 94  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23]]
```

Figure 5: Confusion Matrix obtained during Classification

TABLE 1 summarizes the result of the classification showing the loss percentage and the accuracy percentage. It is observed that the Inception module of CNN model achieves significant detection performance as the average accuracy is 98.08%.

Table 1: Result of the Classification

Metrics	Accuracy	In Percentage (%)
1.	Test loss	7.55130105622482
2.	Test accuracy	97.30784237222005
3.	Kappa accuracy	96.92939627277772
4.	Overall accuracy	97.30784237222005
5.	Average accuracy	98.0812132164497

The Classification map for the ground truth image (Figure 3) which is designated into 16 classes is obtained as shown in Figure 6.



Figure 6: Classification map obtained after applying classification model on Indian pines data set

5. Conclusion and Future Scope

In this paper, an Inception module architecture of CNN model using deep learning tools is implemented for the classification task. Inception module convolves the input tensors with multiple number of filters and connects the obtained results into one block which results in higher performance of the classification model. This approach systematically constructs superior features that encodes pixel's spectral and spatial details and presents high-level performance for every hyperspectral satellite image data. Although PCA has done a good job in case of dimensionality reduction, there are other methods which can be implemented to yield better accuracy depending upon the use case. The above approach can be used in performing classification of satellite images of a particular area considered for study to observe the different resources in that area such as agricultural land, water, built-up areas, etc.

As a part of future work, we would like to gather the dataset of a particular region for a particular time period, say 10 years, on which the discussed classification approach can be applied to the dataset collected for every year and the changes in land use/land cover patterns can be detected. By studying these patterns, observations can be made on following factors such as vegetation changes which will help in taking measures for improving cultivation patterns, change in built-up areas which denotes the rate of urbanization on which several actions can be taken to control the rate, changes in water covered areas which will help in reducing water scarcity. This land use/land cover pattern describes the impact of human activities on the environment which plays a major role in the planning and development of a region.

References

- [1] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, "Going Deeper With Convolutions" (IEEE) 2015.

- [2] Lokman, G., & Yilmaz, G. "Hyperspectral image classification using Support Vector Neural Network algorithm", 7th International Conference on Recent Advances in Space Technologies (RAST) 2015.
- [3] Ido Faran, Nathan S. Netanyahu, Maxim Shoshany, Fadi Kizel, Eli (Omid) David, Jisung Geba Chang, Ronit Rud, "Ground Truth Simulation For Deep Learning Classification Of Mid-Resolution Venus Images Via Unmixing Of High-Resolution Hyperspectral Fenix Data", IGARSS 2019, IEEE International Geoscience and Remote Sensing Symposium 2019.
- [4] Liu, B., Yu, X., Zhang, P., Yu, A., Fu, Q., & Wei, X, "Supervised Deep Feature Extraction for Hyperspectral Image Classification". IEEE Transactions on Geoscience and Remote Sensing, 56(4), 1909–1921. (2018).
- [5] Amir Ghaderi, Vassilis Athitsos, "Selective Unsupervised Feature Learning With Convolutional Neural Network (S-CNN)" 2016.
- [6] Lu, Y., Perez, D., Dao, M., Kwan, C., & Li, J., "Deep Learning with Synthetic Hyperspectral Images for Improved Soil Detection in Multispectral Imagery", 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON) 2018.
- [7] Shen, Y., Xiao, L., Chen, J., & Pan, D., "A Spectral-Spatial Domain-Specific Convolutional Deep Extreme Learning Machine for Supervised Hyperspectral Image Classification", IEEE Access, 2019.
- [8] K. Makantasis, K. Karantzalos, A. Doulamis and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, 2015.
- [9] H. S. Obaid, S. A. Dheyab and S. S. Sabry, "The Impact of Data Pre-Processing Techniques and Dimensionality Reduction on the Accuracy of Machine Learning." 2019 9th Annual Information Technology, Electromechanical Engineering and Microelectronics Conference (IEMECON), Jaipur, India, 2019.
- [10] Fan Wang, Rong Zhang and Qian Wu. "Hyperspectral Image Classification Based On PCA Network," 2016 8th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS) 2019.
- [11] Selin Bostan, Mehmet Akif Ortak, Caglayan Tuna, Alper Akoguz, Elif Sertel and Burak Berk Ustundag. "Comparison of Classification Accuracy of Co-located Hyperspectral & Multispectral Images for Agricultural Purposes", Fifth International Conference on Agro-Geoinformatics 2016.
- [12] Feng Zhou, Renlong Hang, Qingshan Liu and Xiaotong Yuan. "Pyramid Fully Convolutional Network for Hyperspectral and Multispectral Image Fusion" IEEE 2019.
- [13] Hao Chen and C.H Chen, Univ of Massachusetts Dartmouth. "Hyperspectral Image Data Unsupervised Classification Using Gauss-Markov Random Fields and PCA Principle", International Geoscience and Remote Sensing Symposium IEEE 2002.
- [14] Liren Zhang, Hao Lu, Hang Zhang, Yuefeng Zhao, Huaqiang Xu and Jingjing Wang. "Hyperspectral Characteristics in Support of Object Classification and Verification", IEEE Access 2019.
- [15] Aamir Naveed Abbasi, Mingyi He, "Convolutional Neural Network with PCA and Batch Normalization for Hyperspectral Image Classification" IGARSS 2019, IEEE International Geoscience and Remote Sensing Symposium 2019.